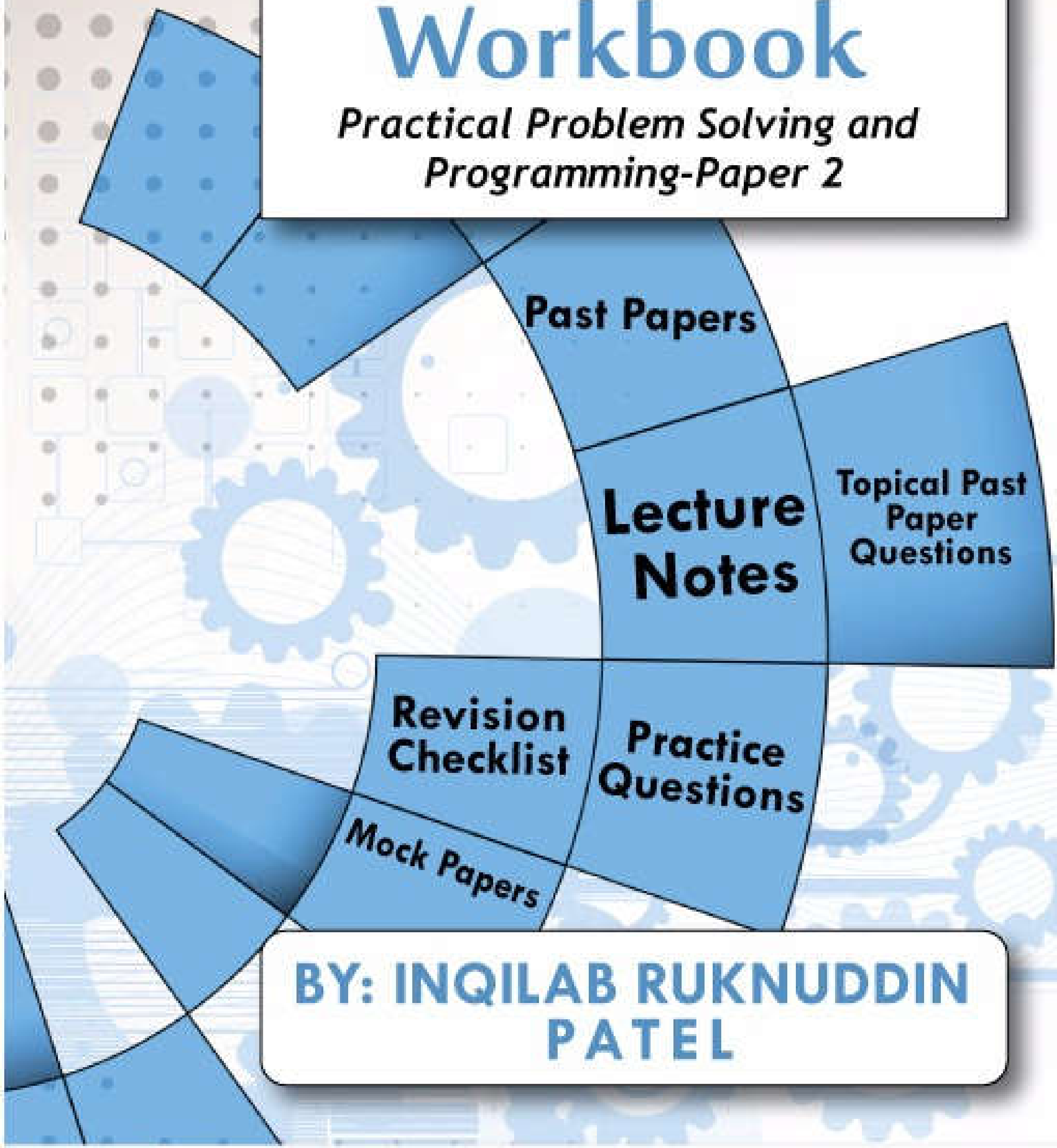
Workbook

Practical Problem Solving and Programming-Paper 2



Past Papers

Lecture Notes

Topical Past Paper Questions

Revision Checklist

Practice Questions

Mock Papers

BY: INQILAB RUKNUDDIN PATEL

O LEVEL & IGCSE

COMPUTER *SCIENCE*

INQILAB PATEL

+923002724734 /inqilabpatel

www.ruknuddin.com

CONGRATULATIONS

MUHAMMAD ANAS GADAR

FOR ACHIEVING

TOP IN WORLD POSITION

IN COMPUTER SCIENCE

O Level Summer 2017 exam

THE CITY SCHOOL PAF CHAPTER

DESIGN BY MAAZ SHAHID

Contents

Syllabus content & assessment at a glance 4

About the developer of this workbook 5

Disclaimer 6

Introduction 8

Algorithm 16

Structure Chart 27

Flowchart 32

Finding Output from flowchart 74

Algorithm pseudo code 110

Errors in Pseudo code and Program 170

Finding output from pseudo code 192

Database 198

Introduction to Logo 233

2.2 Programming 239

Pre-Release Materials 260

Syllabus content & assessment at a glance

Sections	Topics
Section 1	**Theory of Computer Science** 1.1 Data representation 1.1.1 Binary systems 1.1.2 Hexadecimal 1.1.3 Data storage 1.2 Communication and Internet technologies 1.2.1 Data transmission 1.2.2 Security aspects 1.2.3 Internet principles of operation 1.3 Hardware and software 1.3.1 Logic gates 1.3.2Computer architecture and the fetch-execute cycle 1.3.3 Input devices 1.3.4 Output devices 1.3.5 Memory, storage devices and media 1.3.6 Operating systems 1.3.7High- and low-level languages and their translators 1.4 Security 1.5 Ethics
Section 2	**Practical Problem-solving and Programming** 2.1 Algorithm design and problem-solving 2.1.1 Problem-solving and design 2.1.2 Pseudocode and flowcharts 2.2 Programming 2.2.1 Programming concepts 2.2.2 Data structures; arrays 2.3 Databases

Assessment at a glance

Components	**Weighting**
Paper 1 Theory 1 hour 45 minutes This written paper contains short-answer and structured questions. All questions are compulsory. No calculators are permitted in this paper. 75 marks Externally assessed.	**60%**
Paper 2 Problem-solving and Programming 1 hour 45 minutes This written paper contains short-answer and structured questions. All questions are compulsory. 20 of the marks for this paper are from questions set on the pre-release material. 1 No calculators are permitted in this paper. 50 marks Externally assessed.	**40%**

About the developer of this workbook

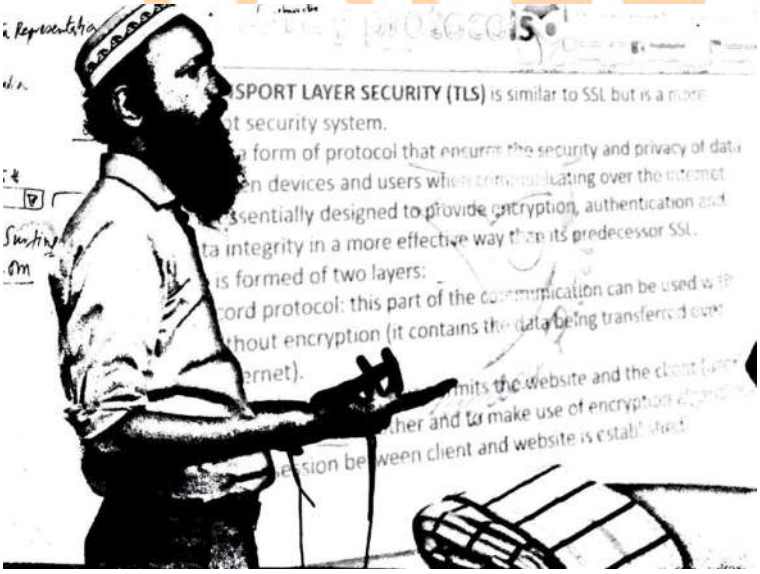
Inqilab Patel is an O &A Level Computer Teacher. He has taught in many schools including Yaqeen Model School, Karachi Cadet School, KN Academy, Beacon House and The City School, **PAF Chapter, Hexis A Level** and **Nakhlah** Boys Campus Society. **Cambridge** has selected him as a **Member** of **Cambridge Editorial Review Board**. He is also associated with Aga **Khan University Examination Board in** the capacity of **Chief Examiner, Item Writer, E-Marker**, **Karachi Board of Secondary Education**the capacity of **Deputy Head Examiner** and **Sindh Board of Technical Education**.

His entire career path revolves around computer science; either he was a student or a teacher. He got a chance to polish his skills of teaching and studying more about computers at various levels which has given him great confidence in presenting himself for any senior level position of transferring his knowledge to the youth.

He has not stopped, he is continuing with his education at the higher levels. It is his second semester of MPhil computer studies from a well-known university of Pakistan; The Institute of Business & Technology.

Inqilab Patel knows a lot of methods of teaching computers and has developed tutorial notes, worksheets and assignments for my students. He also maintains a website (www.inqilabpatel.com) which is specifically designed for the support of those who want to excel in GCSE computer science. He also regularly contributes material to CIE teacher support website, for which he receives appreciation from different people across the world.

He has also received various training in innovative and special methods of teaching this subject.


SPORT LAYER SECURITY (TLS) is similar to SSL but is a more
t security system.
form of protocol that ensures the security and privacy of data
n devices and users when communicating over the internet
ssentially designed to provide encryption, authentication and
ta integrity in a more effective way than its predecessor SSL.
is formed of two layers:
ord protocol: this part of the communication can be used with
thout encryption (it contains the data being transferred over
ernet).

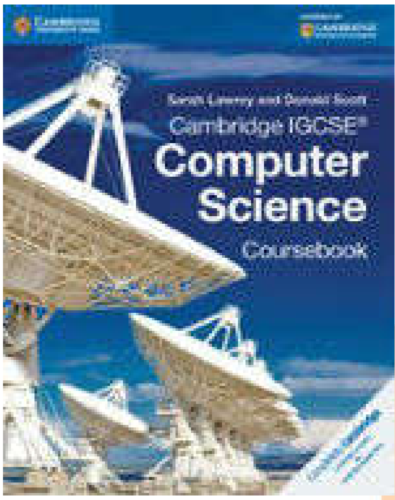
Disclaimer

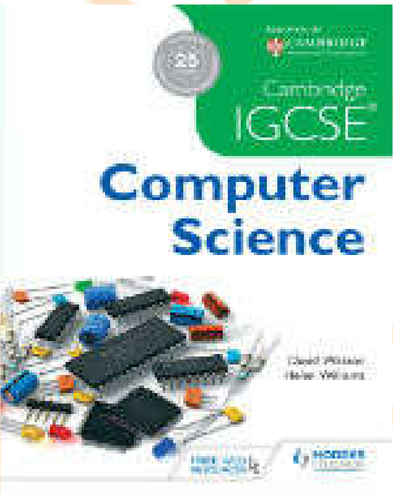
This workbook is developed by combining different materials related to Cambridge IGCSE & O Level Computer Science 0478 & 2210. It is combination of work developed by me and the resources, which are available in different web sites, books, magazines, past papers and guides, just to facilitate students and teachers in preparation for examinations.

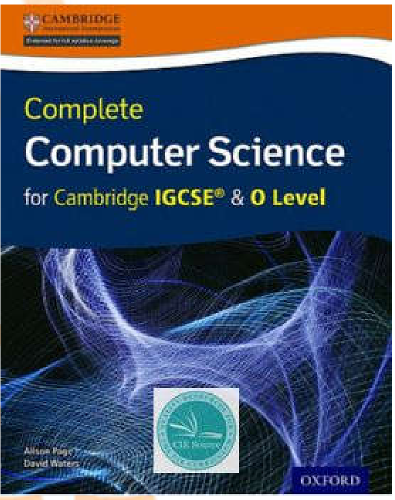
Examination questions and marking schemes used in this workbook are taken from CIE (Cambridge International Examinations)

Note

Study of at least one of the following books, is compulsory for solving this workbook.







Paper 2

Practical Problem-solving andProgramming

Chapter 8

Introduction

2.1.1 Problem-solving and design

Revision Check List (Based on CIE Syllabus)	
Show understanding that every computer system is made up of sub-systems, which in turn are made up of further sub-systems	
Use top-down design, structure diagrams, flowcharts, pseudo code, library routines and subroutines	
Work out the purpose of a given algorithm	
Explain standard methods of solution	
Suggest and apply suitable test data	
Understand the need for validation and verification checks to be made on input data (validation could include range checks, length checks, type checks and check digits)	
Use trace tables to find the value of variables at each step in an algorithm	
Identify errors in given algorithms and suggest ways of removing these errors	
Produce an algorithm for a given problem (either in the form of pseudo code or flowchart)	
Comment on the effectiveness of a given solution	

A system is a combination of parts or components, which work together to control a task or activity.

Computer System

A system which is made up of software, data, hardware, communications and people is considered as computer system.

Each computer system can be divided up into a set of sub-systems. Each subsystem can be further divided into sub-systems and so on until each sub-system just performs a single action.

Computer system is often divided up into sub-systems. This division can be shown using top-down design to produce structure diagrams that demonstrate the modular construction of the system. Each sub-system can be developed by a programmer as sub-routine or an existing library routine may be already available for use. How each sub-routine works can be shown by using flowcharts or pseudo code.

- Top-down design
- Structure diagrams
- Flowcharts
- Pseudo code
- Library routines
- Sub-routines

1. Top-Down Design

Top-down design is the breaking down of a computer system into a set of subsystems, then breaking each sub-system down into a set of smaller sub-systems, until each sub-system just performs a single action.

This is an effective way of designing a computer system to provide a solution to a problem, since each part of the problem is broken down into smaller more manageable problems. The process of breaking down into smaller sub-systems is called 'stepwise refinement'. This structured approach works for the development of both large and small computer systems. When large computer systems are being developed this means that several programmers can work independently to develop and test different subsystems for the same system at the same time. This reduces the development and testing time.

2. Structure Diagrams

The **STRUCTURE DIAGRAM** shows the design of a computer system in a hierarchical way, with each level giving a more detailed breakdown of the system into sub-systems.

3. Flowcharts

A **FLOWCHART** shows diagrammatically the steps required for a task (sub-system) and the order that they are to be performed. These steps together with the order are called an **ALGORITHM**. Flowcharts are an effective way to communicate the algorithm that shows how a system or sub-system works.

4. Pseudo code

PSEUDO CODE is a simple method of showing an algorithm, using English-like words and mathematical operators that are set out to look like a program.

5. Library routines

A **LIBRARY ROUTINE** is a set of programming instructions for a given task that is already available for use. It is pre-tested and usually performs a task that is frequently required. For example, the task 'get time' in the checking-for-the-alarm-time algorithm would probably be readily available as a library routine.

6. Sub-routines

A **SUB-ROUTINE** is a set of programming instructions for a given task that forms a subsystem, not the whole system. Sub-routines written in high-level programming languages are called 'procedures' or 'functions' depending on how they are used.

Test Data

Test data is the data that is used in tests of a software system.

In order to test a software application we need to enter some data for testing most of the features. Any such specifically identified data which is used in tests is known as test data.

There are following three types of test data:

- Normal Data
- Abnormal Data
- Extreme Data

1. Normal Data

This is the data a computer system should work on.Testing needs to be done to prove that the solution works correctly. In order to do this a set of test data should be used together with the result(s) that are expected from that data. The type of test data used to do this is called **NORMAL DATA**, this should be used to work through the solution to find the actual result(s) and see if these are the same as the expected result(s).

For example, here is a set of normal test data for an algorithm to record the percentage marks from 10 end-of-term examinations for a student and find their average mark:

Normal test data: 50, 50, 50, 50, 50, 50, 50, 50, 50, 50

Expected result: 50

2. Abnormal/Erroneous Data

This is data that should cause the system to tell the user that there is a problem with data entered into the system. Testing also needs to be done to prove that the solution does not give incorrect results. In order to do this, test data should be used that will be rejected as the values are not suitable. This type of test data is called **ERRONEOUS** or **ABNORMAL TESTDATA**; it should be rejected by the solution.

For example erroneous/abnormal data for an algorithm to record the percentage marks from 10 end-of-term examinations for a student and find their average mark could be:

Erroneous/abnormal data:–12, eleven

Expected results: these values should be rejected

3. Extreme Data

When testing algorithms with numerical values, sometimes only a given range of values should be allowed. For example, percentage marks should only be in the range 0 to 100. The algorithm should be tested with **EXTREME DATA**, which, in this case, are the largest and smallest marks that should be accepted. Extreme data are the largest and smallest values that normal data can take.

Extreme data: 0, 100

Expected results: these values should be accepted

4. Boundary Data

This is used to establish where the largest and smallest values occur. For example, for percentage marks in the range 0 to 100, the algorithm should be tested with the following boundary data; at each boundary two values are required, one value is accepted and the other value is rejected.

Boundary data for 0 is –1, 0

Expected results: –1 is rejected, 0 is accepted

Rogue Values

A sequence of inputs may continue until a specific value is input. This value is called a **rogue value** and must be a value which would not normally arise.

A rogue value lets the computer know that a sequence of input values has come to an end.

Example

A number of marks are to be input (terminated by a rogue value of -1). How many of them are over 50?

```
Counter ←0
INPUT Marks
REPEAT
    IF Marks >50 THEN Above50 ←Above50 +1
    INPUT Marks
UNTIL Marks=-1
OUTPUT Above50
```

Validation and Verification

Validation and verification are two ways to check that the data entered into a computer is correct. Data entered incorrectly is of little use.

Data verification

Verification is performed to ensure that the data entered exactly matches the original source. Verification means checking the input data with the original data to make sure that there have been no transcription errors (transcription means copying the data). The standard way to do this is to input the data twice to the computer system. The computer then checks the two data values (which should be the same) and, if they are different, the computer knows that one of the inputs is wrong. E.g. entering password twice during sig-up.

Verification methods include:

- double entry
- screen/visual check (proof reading)
- parity check
- Checksum.

Validation is an automatic computer check to ensure that the data entered is sensible and reasonable. It does not check the accuracy of data.

For example, a secondary school student is likely to be aged between 11 and 16. The computer can be programmed only to accept numbers between 11 and 16. This is a **range** check.

However, this does not guarantee that the number typed in is correct. For example, a student's age might be 14, but if 11 are entered it will be valid but incorrect.

A validation check is a rule that is built into a database to check that the data entered is:

- Sensible
- Reasonable
- Within acceptable boundaries
- Complete

It does NOT mean that the data is actually correct, that requires verification.

There are a number of different validation rules that can be used in a database:

Type Checks - Field data types provide a basic method of validation. Field data types are assigned to fields during the creation of the database table and data types such as

Numeric, Boolean, Date/Time and Image restrict what can be entered. If a user tries to enter text in a date field or alphabetic characters in a numeric field, their entry will be rejected.

Range checks - these are used to limit the range of data a user can enter. The 'day' part of a date must be in the range 1 to 31. An exam grade should be in the range 'A'...'G' or 'U'.

Check digits - this type of check is used with numbers. An extra 'check digit' is calculated from the numbers to be entered and added to the end. The numbers can then be checked at any stage by recalculating the check digit from the other numbers and seeing if it matches the one entered. One example where a check digit is used is in the 10 digit ISBN number which uniquely identifies books.

The last number of the ISBN is actually the check digit for the other numbers, for example - the ISBN 0192761501.

Presence checks - these simply check that an entry has been made in a particular field i.e. a null value (empty field) is not permitted. Usually, not every field in a record needs to be filled in, however there are likely to be some that must have a value and the presence check means that the system will not allow the record to be saved until an entry is made. An application for a passport must have the applicant's surname.

Length Checks - All alphanumeric data has a length. A single character has a length of 1 and a string of text such as “Hello World” has a length of 11 (spaces are counted in text strings). A length check ensures that such data is either an exact length or does not exceed a specified number of characters. Mobile phone numbers are stored as text and should be 11 characters in length.

Lookup - A lookup check takes the value entered and compares it against a list of values in a separate table. It can then return confirmation of the value entered or a second list based on the value. One use of lookups restricts users to pre-defined input using drop-down lists. A user is forced to use a list box to select from a predetermined list of valid values.

Summary of validation

Validation type	How it works	Example usage
Range check	Checks that a value falls within the specified range	Number of hours worked must be less than 50 and more than 0
Length check	Checks the data isn't too short or too long. Values must be a specific length.	A password which needs to be six letters long
Limit Check	Similar to Range Check but the rule involves only one limit.	>=0 means reject negative numbers. Date of birth must not be later than a date.
Type Check	Checks that the data entered is of a given data type,	Number of brothers or sisters would be an integer (whole number).
Character Check	Checks that when a string of characters is entered it does not contain any invalid characters or symbols,	A name would not contain characters such as %, and a telephone number would only contain digits or (,), and+.

Validation type	How it works	Example usage
Format Check	Checks the data is in the right format. Values must conform to a specific pattern, for example, two letters followed by six digits followed by a single letter	A National Insurance number is in the form LL 99 99 99 L where L is any letter and 9 is any number
Presence check	Checks that data has been entered into a field	In most databases a key field cannot be left blank
Check digit	The last one in a code are used to check the other digits are correct	Bar code readers in supermarkets use check digits

length check – e.g. only 30 characters in name field
character check – e.g. name doesn't contain numeric chars
range check – e.g. day of month in date is between 1 and 31
format check – e.g. date in the form xx/yy/zz
check digit – e.g. end digit on bar code to check if it is valid
type check – e.g. integer, real
(presence check = 0)

Q 8.1) Activity of data validation and verification:

1) What is an automatic computer check to make sure data entered is sensible and reasonable known as?

a) Double entry b) Verification c) Validation

2) What validation type would make sure a post code was entered in the correct format?

a) Length check b) Format Check c) Presence check

3) What validation type would you use to check that numbers fell within a certain range?

a) Range check b) Presence Check c) Check digit

4) What validation type checks that a field is not left blank?

a) Format check b) Length check c) Presence check

5) What validation type uses the last one or two digits to check the other digits are correct?

a) Length check b) Format check c) Check digit

6) What validation type checks a minimum number of characters have been entered?

a) Length check b) Format check c) Range check

7) Data is to be entered into a computer in the format YYMMDD. Which of the following is not a valid date?

a) 310921 b) 211113 c) 21st June 2004

8) Which of the following statements is false?

a) Validation can check that the data is sensible
b) Validation can check that the data falls between certain allowable boundaries
c) Validation can check that the data is correct

9) Which of the following is NOT a method of verification?
a) Double entry - typing the data in twice and getting the computer to check the second version against the first
b) Using presence, range and length checks to make sure that no mistakes happen
c) Printing out what you have typed in and comparing it against the source data

Q8.2) Summer 2014 pq11

A hospital holds records of its patients in a database. Four of the fields are:

- date of visit (dd/mm/yyyy)
- patient's height (m)
- 8-digit patient ID
- contact telephone number

The presence check is one possible type of validation check on the data. For each field, give another validation check that can be performed. Give an example of data which would fail your named validation check. A different validation check needs to be given for each field.

field name	**name of validation check**	**example of data which would fail the validation check**
date of visit		
patient's height		
patient ID		
Contact telephone number		

Marking scheme

Field Name	*Name of validation check*	*Example of data which would fail the validation check*
Date of visit	*format check* *type/character check*	*e.g. 2012/12/04* *e.g. 3rd March 2012*
Patient's height	*range check* *limit check*	*can't be < 0 or > 2.5m* *e.g. –5, five* *e.g. 8, -3,*
Patient ID	*type check* *length check* *range check*	*(can't be < 0 or > 99999999)* *e.g. 3142ABCD* *e.g. 2131451, 136498207* *e.g. –3, 851341625*
Contact telephone number	*length check* *type/character check* *format check*	*e.g. 0773141621834* *e.g. 7H215GD* *e.g. 01223/123456/8901234*

Q8.3) Summer 2013 P12

A company requests new customers who register online to give the following details:

- Name
- address
- Type of credit/debit card
- Payment card number

All details must be entered.

(a) (i) Describe one suitable different validation check for each field.

Name: ………………………………………………………………………………………..

Address: ……………………………………………………………………………………..

type of credit/debit card: ……………………………………………………………………

payment card number: …………………………………………………………………… [4]

Q8.4) Summer 2012 P12

State two different validation checks and give an example of their use. Each example should be different.

Check 1: ………………………………………………………………………………………

Use: ……………………………………………………………………………………………

Check 2: ………………………………………………………………………………………

Use: …………………………………………………………………………………………… [4]

Chapter 9

Algorithm

2.1.2 Algorithm Pseudo code

An algorithm is a series of well defined steps which gives a procedure for solving a type of problem.
The word algorithm comes from the name of 9th century mathematician al-Khwarizmi (Muhammad Bin Musa Al-Khwarizmi).
In fact, even the word algebra is derived from his book "Hisab al-jebrw'al-muqabala"

2.1.2 Pseudo code
- understand and use pseudo code for assignment, using ←
- understand and use pseudo code, using the following conditional statements:

IF … THEN … ELSE … ENDIF
CASE … OF … OTHERWISE … ENDCASE
- understand and use pseudo code, using the following loop structures:

FOR … TO … NEXT
REPEAT … UNTIL
WHILE … DO … ENDWHILE
- understand and use pseudo code, using the following commands and statements:

INPUT and OUTPUT (e.g. READ and PRINT)
totalling (e.g. Sum ← Sum + Number)
counting (e.g. Count ← Count + 1)
(Candidates are advised to try out solutions to a variety of different problems on a computer using a language of their choice; no particular programming language will be assumed in this syllabus.)

"An **algorithm** is a sequence of steps for a computer program to accomplish a task."

In general, an 'algorithm' is the name given to a defined set of steps used to complete a task.

For instance you could define an algorithm to make a cup of tea. You start by filling the kettle, and then place a tea bag in the cup and so on.

In computer terms, an algorithm describes the set of steps needed to carry out a software task. This mini-web takes you through the topic of algorithm

The concept of a program

A program is a sequence of instructions or programming language statements written to make a computer perform certain tasks.

Well-structured programs require a programming language to support the following program constructs:

1. Sequence
2. Assignment
3. selection
4. iteration

A computer's processor can only run a computer program in the form of a file of machine code, which is a sequence of binary codes representing instructions for the processor.

The instruction set for a family of processors is the machine language in which machine code is written for that family of processors.

When machine code runs, the processor repeatedly:

- Fetches an instruction from internal memory
- Decodes the instruction
- Executes the instruction.

Pseudo Code Guide

Atomic type names

The following keywords are used to designate atomic data types:

INTEGER:	A whole number (without fractional part)
REAL:	A number capable of containing a fractional part
CHAR:	A single character
STRING:	A sequence of zero or more characters
BOOLEAN:	The logical values TRUE and FALSE
DATE:	A valid calendar date

Literals

Literals of the above data types are written as follows:

Integers: Written as normal in the denary system, e.g. 5, -3

Real: Always written with at least one digit on either side of the decimal point, zeros being added if necessary, e.g. 4.7, 0.3, -4.0, 0.0

Char: A single character delimited by single quotes, e.g. 'x', 'C', '@'

String: Delimited by double quotes. A string may contain no characters (i.e. the empty string) e.g. "This is a string", ""

Boolean: TRUE, FALSE

Variable:

Variable is memory location where a value can be stored. The values stored in a variable are changed during execution.

Identifiers

Identifiers (the names given to variables, constants, procedures and functions) are in mix case. They can only contain letters (A–Z, a–z) and digits (0–9). They must start with a letter and not a digit. Accented letters and other characters, including the underscore, should not be used.

As in programming, it is good practice to use identifier names that describe the variable, procedure or function they refer to. Single letters may be used where these are conventional (such as i and j when dealing with array indices, or X and Y when dealing with coordinates) as these are made clear by the convention.

Keywords identified elsewhere in this guide should never be used as variables.

Identifiers should be considered case insensitive, for example, Countdown and Countdown should not be used as separate variables.

Assignments

Storing values in a variable is known as assignment .

The assignment operator is ←.

Assignments should be made in the following format:

<identifier> ← <value>

<u>For example:</u>

```
Counter ← 0
Counter ← Counter + 1
TotalToPay ← NumberOfHours * HourlyRate
```

Variable declarations

It is good practice to declare variables explicitly in pseudo code.

Declarations are made as follows:

```
    DECLARE<identifier> : <data type>
```

Example

```
    DECLARE Surname : STRING
    DECLARE FirstName : STRING
    DECLARE DateOfBirth : DATE
    DECLARE Section : CHAR
    DECLARE Counter : INTEGER
    DECLARE TotalToPay : REAL
    DECLARE GameOver : BOOLEAN
```

Constant:

Constant is memory location where a value can be stored but the stored value remaining same during execution.

It is good practice to use constants if this makes the pseudo code more readable, as an identifier is more meaningful in many cases than a literal. It also makes the pseudo code easier to update if the value of the constant changes.

Constant declaration

Constants are normally declared at the beginning of a piece of pseudo code (unless it is desirable to restrict the scope of the constant).

Constants are declared by stating the identifier and the literal value in the following format:

```
CONSTANT<identifier> = <value>
```

Example

```
CONSTANT HourlyRate = 6.50
CONSTANT DefaultText = “N/A”
```

Only literals can be used as the value of a constant. A variable, another constant or an expression must never be used.

Input and output

Values are input using the INPUT command as follows:

```
INPUT <identifier>
```

The identifier should be a variable (that may be an individual element of a data structure such as an array, ora custom data type).

Values are output using the OUTPUT command as follows:

```
OUTPUT <value(s)>
```

Several values, separated by commas, can be output using the same command.

Example – INPUT and OUTPUT statements

```
INPUT Answer
OUTPUT Score
OUTPUT "You have ", Lives, " lives left"
```

Note that the syllabus for IGCSE (0478) gives READ and PRINT as examples for INPUT and OUTPUT, respectively.

Arithmetic operations

Standard arithmetic operator symbols are used:

- + Addition
- - Subtraction
- * Multiplication
- / Division

Care should be taken with the division operation: the resulting value should be of data type REAL, even if the operands are integers.

The integer division operators MOD and DIV can be used. However, their use should be explained explicitly and not assumed.

Multiplication and division have higher precedence over addition and subtraction (this is the normal mathematical convention). However, it is good practice to make the order of operations in complex expressions explicit by using parentheses.

Logic operators

The only logic operators (also called relational operators) used are AND, OR and NOT. The operands and results of these operations are always of data type BOOLEAN.
In complex expressions it is advisable to use parentheses to make the order of operations explicit.

Counting

It is sometimes necessary to count how many times something happens.
To count up or increment by 1, we can use statements such as:

```
Count ← Count + 1
INCREMENT Count by 1
```

Totaling

To keep a running total, we can use a variable such as Total or Sum to hold the running total and assignment statements such as:

```
Total ← Total + Number
ADD Number to Total
```

Comments

Comments are preceded by two forward slashes // . The comment continues until the end of the line. Formulti-line comments, each line is preceded by //.
Normally the comment is on a separate line before, and at the same level of indentation as, the code it refersto. Occasionally, however, a short comment that refers to a single line may be at the end of the line to whichit refers.

Example – comments

```
// This is example of comments
// swapping values of X and Y
Temp ← X // temporarily store X
X ← Y
Y ← Temp
```

Arrays

Arrays are data structure used to store multiple values in a single variable.

Arrays are considered to be fixed-length structures of elements of identical data type, accessible by consecutive index (subscript) numbers. It is good practice to explicitly state what the lower bound of the array (i.e. the index of the first element) is because this defaults to either 0 or 1 in different systems. **Generally, a lower bound of 1 will be used**.

Square brackets are used to indicate the array indices.

Each element in the array is identified using its **subscript** or **index number**. The largest and smallest index numbers are called the *upper bound* and *lower bound* of the array.

One- and two-dimensional arrays are declared as follows (where l, l1, l2 are lower bounds and u, u1, u2 are upper bounds):

simply <identifier >[<l>:<u>]

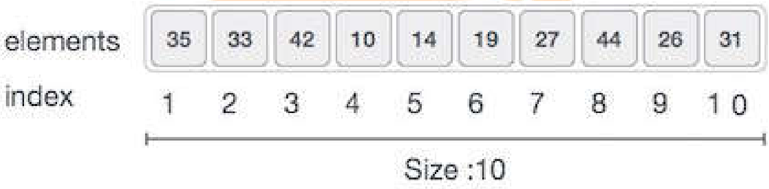
Example

StudentNames[1:30]

For illustration, let's take array declaration to store marks of 10 students.

Marks[1:10]

After storing values in array



elements	35	33	42	10	14	19	27	44	26	31
index	1	2	3	4	5	6	7	8	9	10

Size :10

As per the above illustration, following are the important points to be considered.

- Index starts with 1.
- Array length is 10 which means it can store 10 elements.
- Each element can be accessed via its index. For example, we can fetch an element at index 6 as 19.

Using arrays

In the main pseudo code statements, only one index value is used for each dimension in the square brackets.

Example

```
DECLARE StudentNames[1:30] : STRING
StudentNames[1]  ← "Abdullah"
```

StudentsNames

1	Abdullah
2	Rumaisa
3	Rashid
4	Afeera
5	Laiba
6	Patel
7	Smith
29	Mani
30	Muzna

Selection:

For selection following statements are used:

- IF
- CASE

IF statements

IFstatements is used when there are two or one options.

When there is only one option IF statements without an ELSE clause is written as follows:

```
IF<condition>THEN
        <statements if true>
ENDIF
```

Example

```
IF Number>Largest THEN
   Largest←Number

ENDIF
```

When there are two options IF statements with an ELSE clause is written as follows:

```
IF <condition>THEN
        <statements if true>
  ELSE
        <statements if false>
ENDIF
```

Example

```
IF Marks>=50 THEN
          Result ← "Pass"
  ELSE
          Result ← "Fail"
ENDIF
```

Note that the THEN and ELSE clauses are only indented by two spaces. (They are, in a sense, a continuation of the IF statement rather than separate statements).

When IF statements are nested, the nesting should continue the indentation of two spaces. In particular, run-on THENIF and ELSE IF lines should be avoided.

CASE statements

CASE is a conditional statement to deal with many possible outcomes.

CASE statements allow one out of several branches of code to be executed, depending on the value of a variable.

CASE statements are written as follows:

```
CASE OF<identifier>
<value 1> : <statement>
<value 2> : <statement>
  ...
ENDCASE
```

AnOTHERWISEclause can be the last case:

```
CASE OF <identifier>
<value 1> : <statement>
<value 2> : <statement>
  ...
  OTHERWISE<statement>
ENDCASE
```

Example – formatted CASE statement

```
INPUT ItemType
CASE OF ItemType
  1: CD ←CD + 1
  2: DVD ←DVD + 1
  3: Video←Video + 1
  4: Book ←Book + 1
  OTHERWISE : Beep
ENDCASE
```

Iteration (Repetition, Loop)

Repetition is used to execute a set of instructions multiple times.
Repetition is also referred as LOOP or ITERATION.
There are following three types of loops:

1. Count-controlled loop
2. Pre-condition loop
3. Post-condition loop

Count-controlled (FOR) loops

Count-controlled loop is used when the number of repetition is already known.

Count-controlled loops are written as follows:

```
FOR <identifier>←<value1> TO <value2>
        <statements>
NEXT<identifier>
```

The identifier must be a variable of data type INTEGER, and the values should be expressions that evaluate to integers.

It is good practice to repeat the identifier after NEXT.

```
FOR <identifier>←<value1> TO <value2> STEP<increment>
        <statements>
NEXT
```

The increment must be an expression that evaluates to an integer. In this case the identifier will be assigned the values from value1 in successive increments of increment until it reaches value2. If it goes past value2, the loop terminates. The increment can be negative.

Example: to input 10 numbers and output their final total

```
Total ← 0
FOR Count ← 1 TO 10
INPUT Number
   Total ←Total + Number
NEXT Count
OUTPUT “The grand total is ”, Total
```

Example: to print 1st 10 even numbers

```
FOR Count ← 1 TO 20STEP 2
      PRINT Count
NEXT Count
```

Pre-condition (WHILE) loops

A loop in which condition is given at the start of loop and which is executed only when the condition is true, is called post-condition loop.

Pre-condition loops are written as follows:

```
WHILE<condition to repeat> DO
        <statements>
ENDWHILE
```

The condition must be an expression that evaluates to a Boolean.

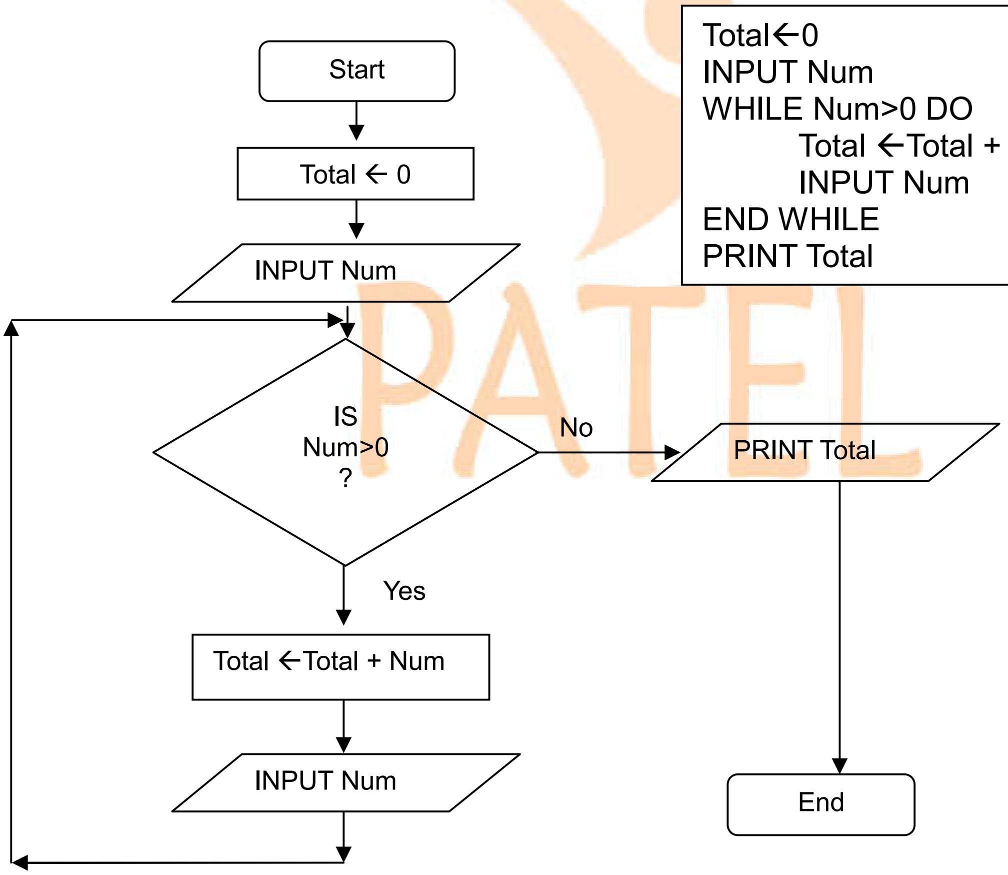
The condition is tested before the statements, and the statements will only be executed if the condition evaluates to TRUE. After the statements have been executed the condition is tested again. The loop terminates when the condition evaluates to FALSE.

The statements will not be executed if, on the first test, the condition evaluates to FALSE.

Example: To input a series of numbers and calculate total and stops if a –ve number is entered:

The condition is checked at the beginning of the loop. If condition is true loop statements are executed again and again.

```
Total←0
INPUT Num
WHILE Num>0 DO
        Total ←Total + Num
        INPUT Num
END WHILE
PRINT Total
```



Post-condition (REPEAT UNTIL) loops

A loop in which condition is given at the end of loop and which is executed only when the condition is false is called post-condition loop.

It is are written as follows:

```
REPEAT
        <Statements>
UNTIL<condition to stop the loop>
```

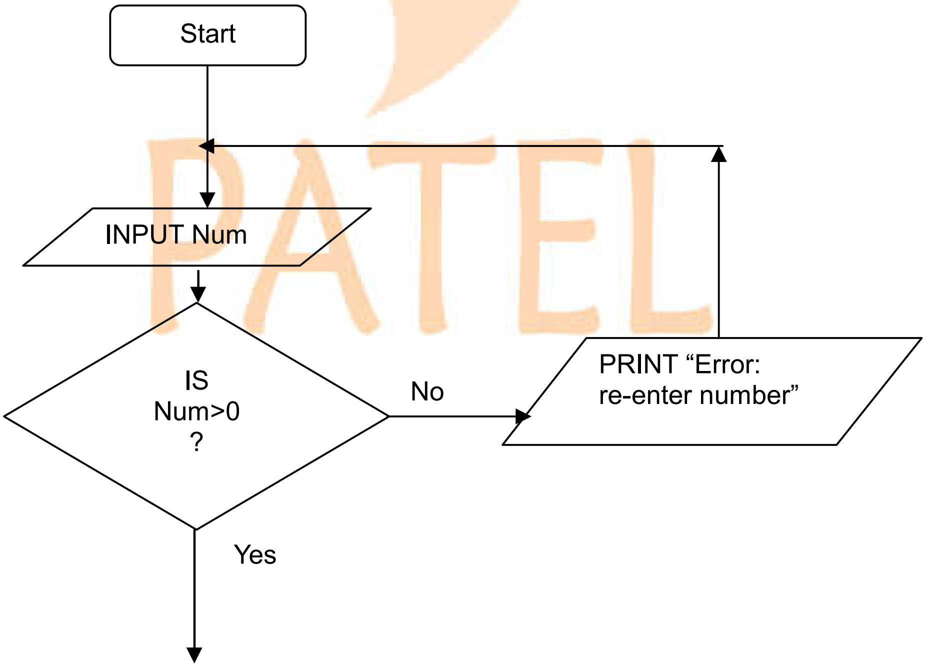
The condition must be an expression that evaluates to a Boolean.

The statements in the loop will be executed at least once. The condition is tested after the statements are executed and if it evaluates to TRUE the loop terminates, otherwise the statements are executed again.

Example: To input and validate a number and to reject it if a negative number is entered and ask to re-enter another number

The condition is checked at the end of the loop. If condition is false loop statements are executed again and again.

```
REPEAT
        INPUT Num
UNTIL Num>0
```



Chapter 10

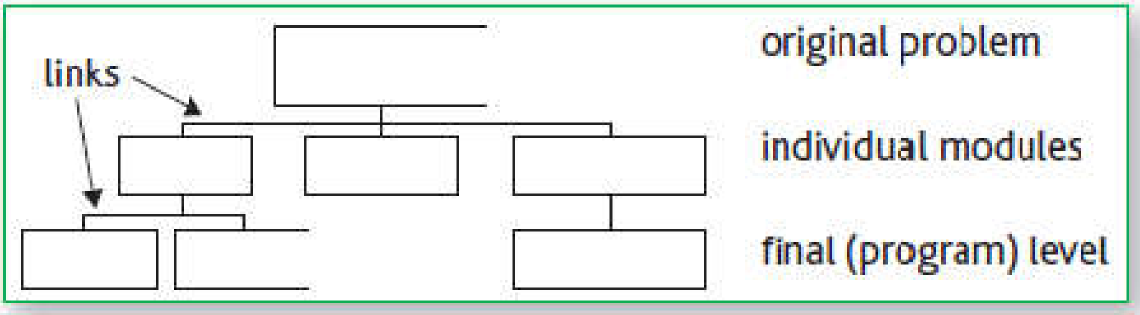
Structure Chart

A Structure Chart in software engineering is a chart which shows the breakdown of a system to its lowest manageable parts. They are used in structured programming to arrange program modules into a tree. Each module is represented by a box, which contains the module's name. The tree structure visualizes the relationships between modules, showing data transfer between modules using arrows.

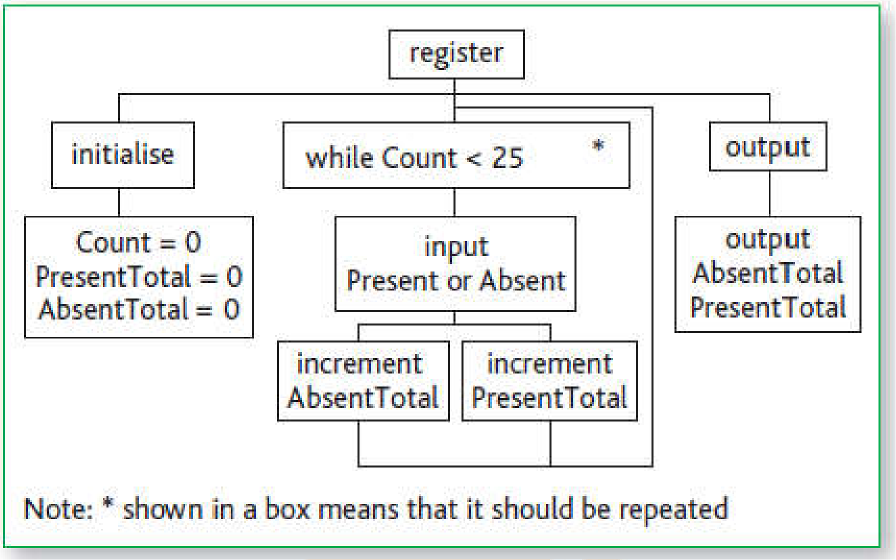
Structured Charts are an example of a **top-down** design where a problem (the program) is broken into its components. The tree shows the relationship between modules, showing data transfer between the models.

Symbol	**Name**	**Meaning**
Module Name	Process	Each Box represents a programming module, this might be something that calculates the average of some figures, or prints out some pay slips
○→	Data Couple	Data being passed from module to module that needs to be processed.
●→	Flag	[Extension - you don't need to know this for the exam] Check data sent to process to stop or start processes. For example when the End of a File that is being read is reached, or a flag to say whether data sent was in the correct format

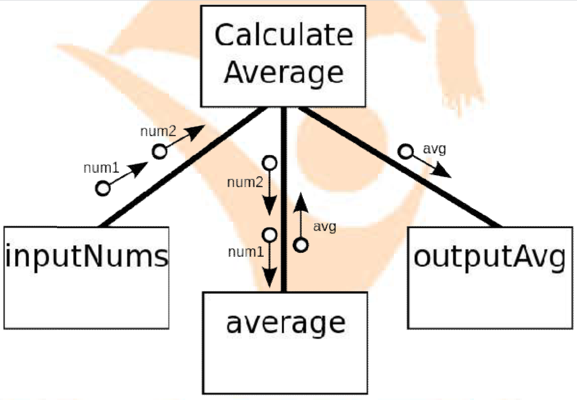
These individual problems can then be solved and combined according to the links that have been used. If the links between the different blocks are used correctly, the result is a solution to the original problem.

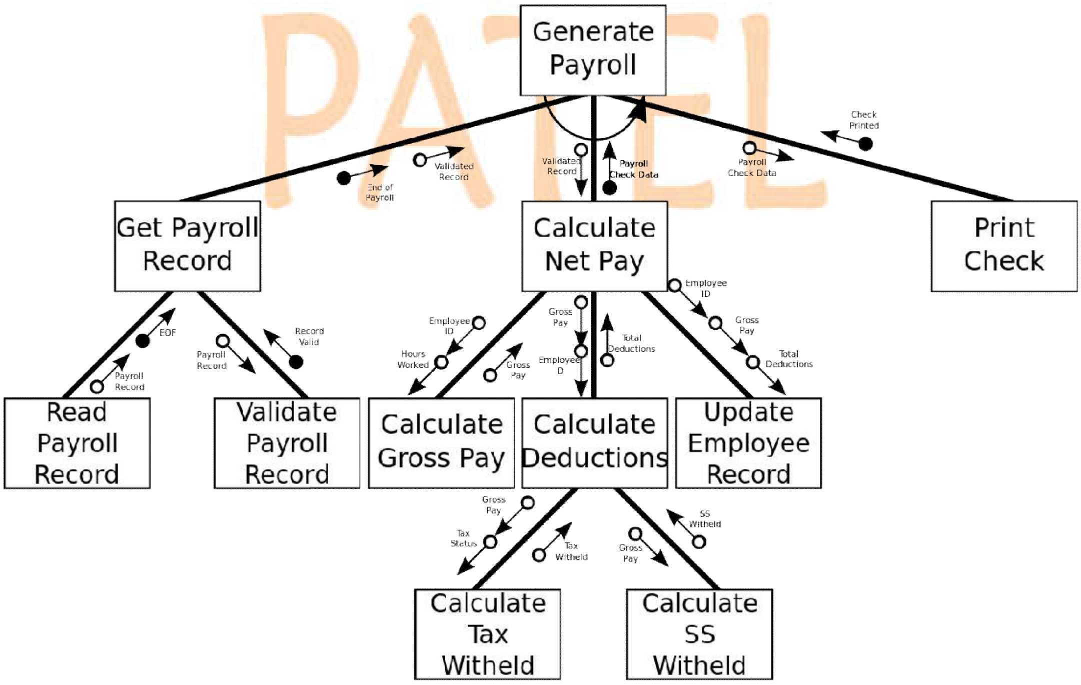


An electronic register is taken of the 25 students in a class. Each student can be either present or absent. The teacher needs to print out the number of students present and the number absent.



Note: * shown in a box means that it should be repeated





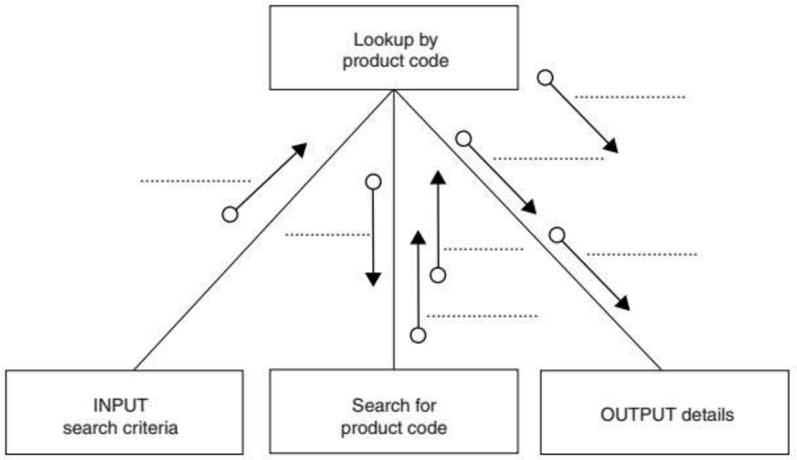
Q10.1) To code the 'Search by product code' procedure, Ahmed draws a structure chart showing the different stages.

The procedure uses the variables shown in the identifier table.

Identifier	Data type	Description
SearchCode	STRING	Product code input by the user
ThisIndex	INTEGER	Array index position for the corresponding product
ThisDescription	STRING	Product description found
ThisRetailPrice	REAL	Product retail price found

You can assume that before the procedure is run, all the product data is read from file PRODUCTS and then stored in three 1D arrays.

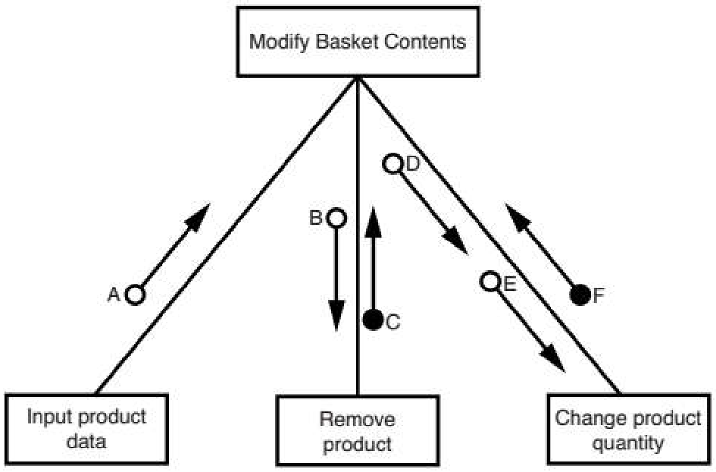
Label the structure chart to show the input(s) and output(s)



[4]

Q10.2) The structure chart shows part of the design of a program for an online shopping system. The user has already added a number of products to their virtual basket.

Draw on the chart, the symbol to show that the process of modifying the basket contents may be iterated (repeated).



[1]

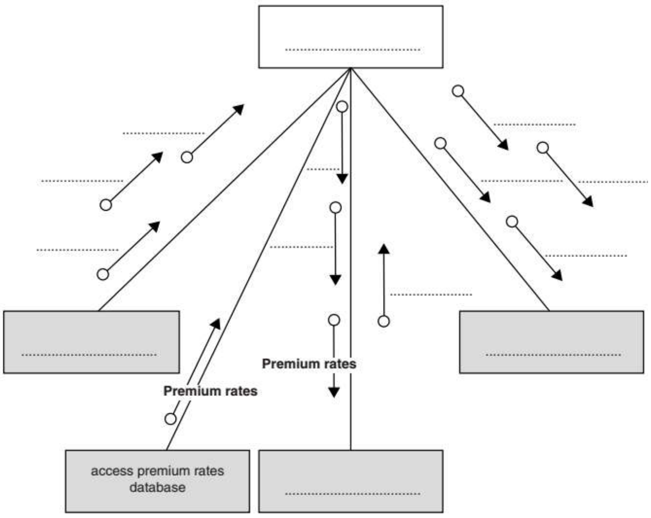
(ii) Each arrow in the structure chart above represents a parameter.
The table below shows the three data items that the six parameters pass between modules. Tick (✓) to match each parameter to the correct data item. [4]

Data item	Parameter					
	A	B	C	D	E	F
Product ID						
Quantity						
Flag Value – indicating operation success or fail						

Q10.3) When the guarantee on a computer runs out, the owner can take out insurance to cover breakdown and repairs.
The price of the insurance is calculated from:

- the model of the computer
- the age of the computer
- the current insurance rates

Following an enquiry to the insurance company, the customer receives a quotation letter with the price of the insurance.
A program is to be produced.
The structure chart below shows the modular design for this process



(a) Using the letters **A** to **D**, add the labeling to the chart boxes on the opposite page.[2]

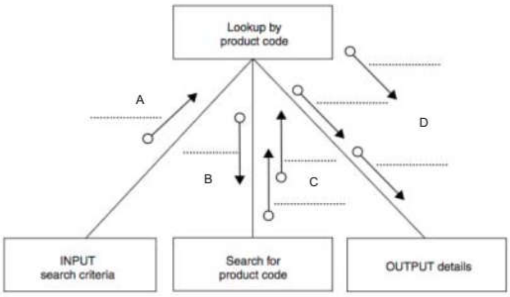
Modules	
A	Send quotation letter
B	Calculate price
C	Produce insurance quotation
D	Input computer details

(b) Using the letters **E** to **J**, complete the labeling on the chart opposite.
Some of these letters will be used more than once.

Data items	
E	CustomerName
F	CustomerEmail
G	Model
H	Age
I	PolicyCharge
J	PolicyNumber

Marking Scheme

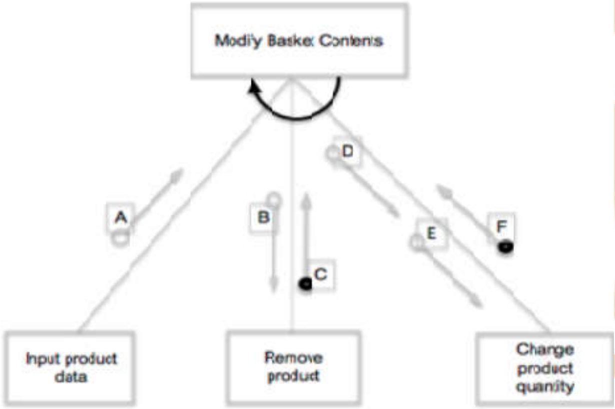
10.1



One mark per group (one or more names) as follows:

A: SearchCode
B: SearchCode // ThisIndex
C: ThisRetailPrice, ThisDescription
D: SearchCode, ThisDescription, ThisRetailPrice [4]

10.2



(ii) [4]

Data Item	Parameter					
	A	B	C	D	E	F
Product ID	✓	✓		✓	(✓)	
Quantity				(✓)	✓	
Flag Value – indicating operation success or fail			✓			✓

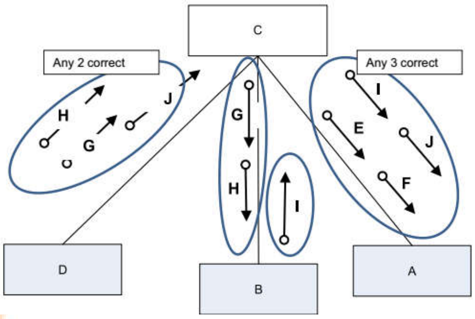
Mark as follows:
Row 1: One mark for tick in A **AND** B, one mark for D **OR** E
Row 2: One mark for D **OR** E (must be opposite of Row 1)
Row 3: One mark for C **AND** F

10.3

3 (a) Control box – C // Produce insurance quotation [1]

D // Input customer details + A // Send quotation letter is correct positions [1]

(b)

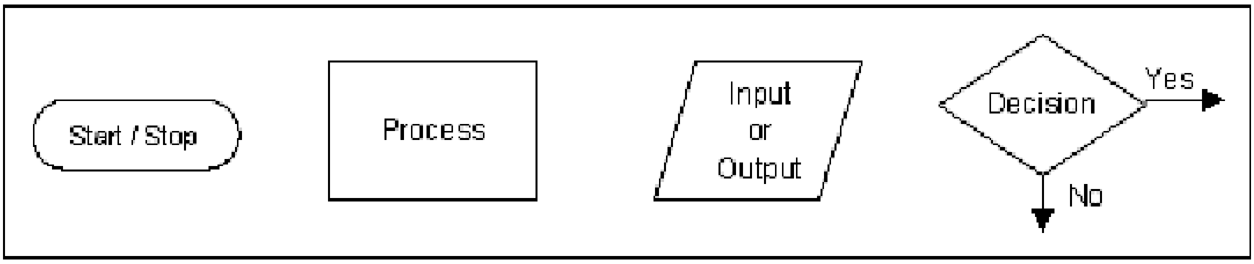


Chapter 11

Flowchart

2.1.2 Flowchart

A flowchart is another way of breaking down a program in the form of a diagram.
The following are recognised flowchart symbols:



Drawing Flowchart

Assignments

Storing values in a variable is called Assignment.
Rectangle symbol is used for assignment and other processing.
The assignment operator is ←.
Assignments should be made in the following format:

<identifier> ← <value>

<identifier> ← <expression>

For example:

Counter ← 0

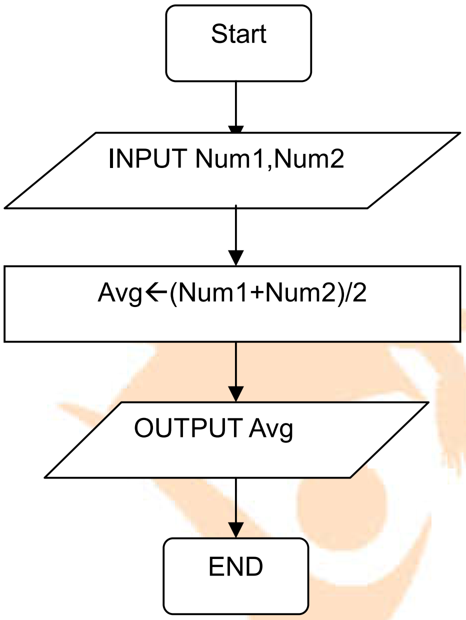
Counter ← Counter + 1

TotalToPay ← NumberOfHours * HourlyRate

Sequence:
Flowchart is drawn in the sequence in which the program is intended to be executed.

Write an algorithm, using flowchart only, which:

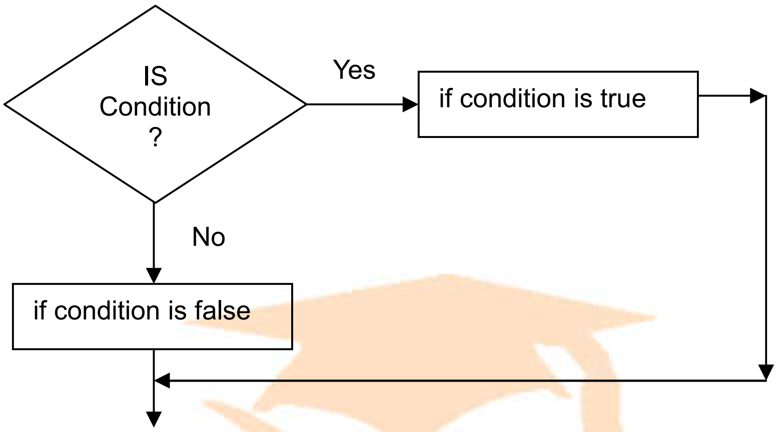
- Inputs two numbers
- Calculate their average
- Output average



Selection:
For selection following statements are used:

- **IS (IF)**
- **CASE**

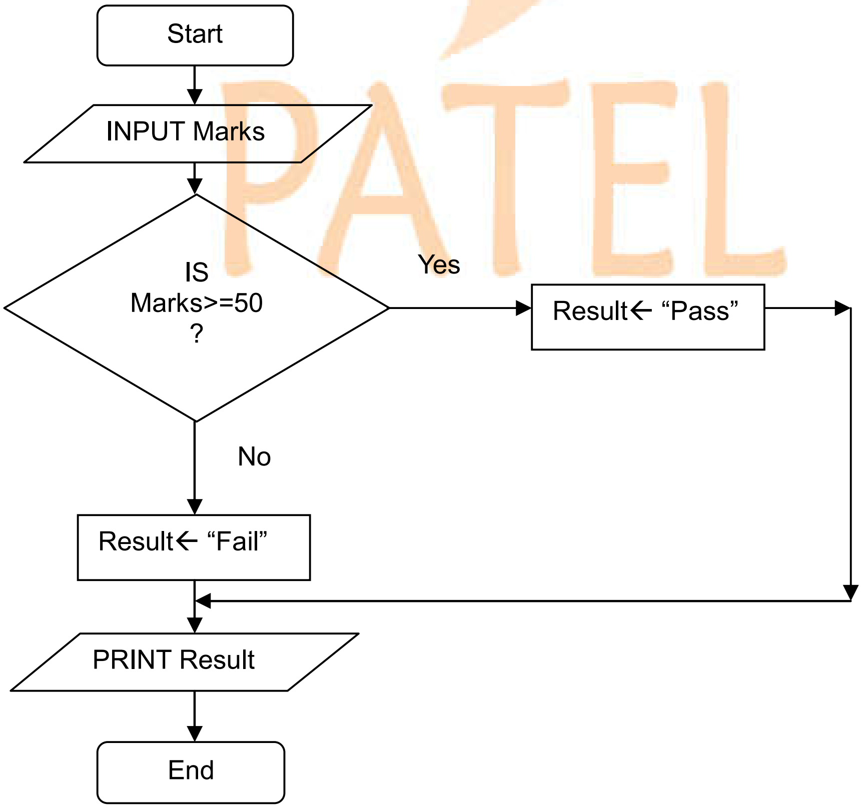
IS (IF) statements
IS statements is used when there are two or one options.
IS (IF) is a conditional statement with different outcomes for true(yes) and false(no).



Example
Write an algorithm, using flowchart only, which:

- inputs marks
- calculate result pass or fail
- output result

(You may assume that result is pass if marks are 50 or above otherwise result is fail)

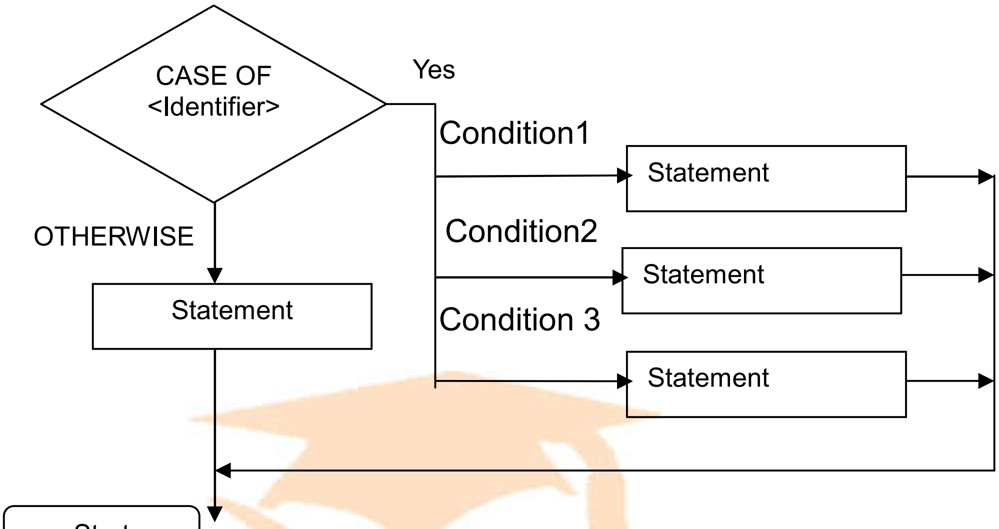


CASE statements

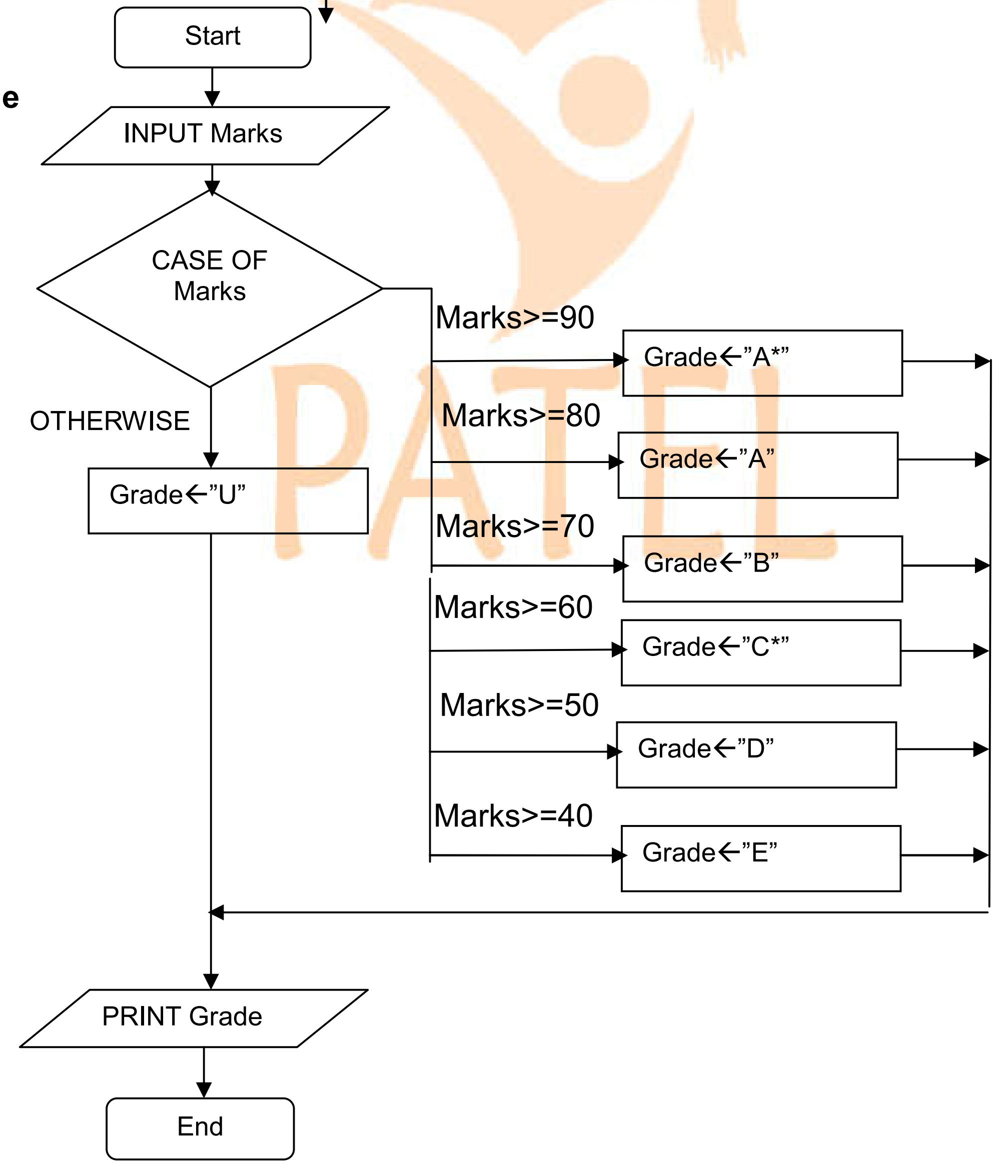
CASE is a conditional statement to deal with many possible outcomes.

CASE statements allow one out of several branches of code to be executed, depending on the value of a variable.

CASE statements are drawn as follows:

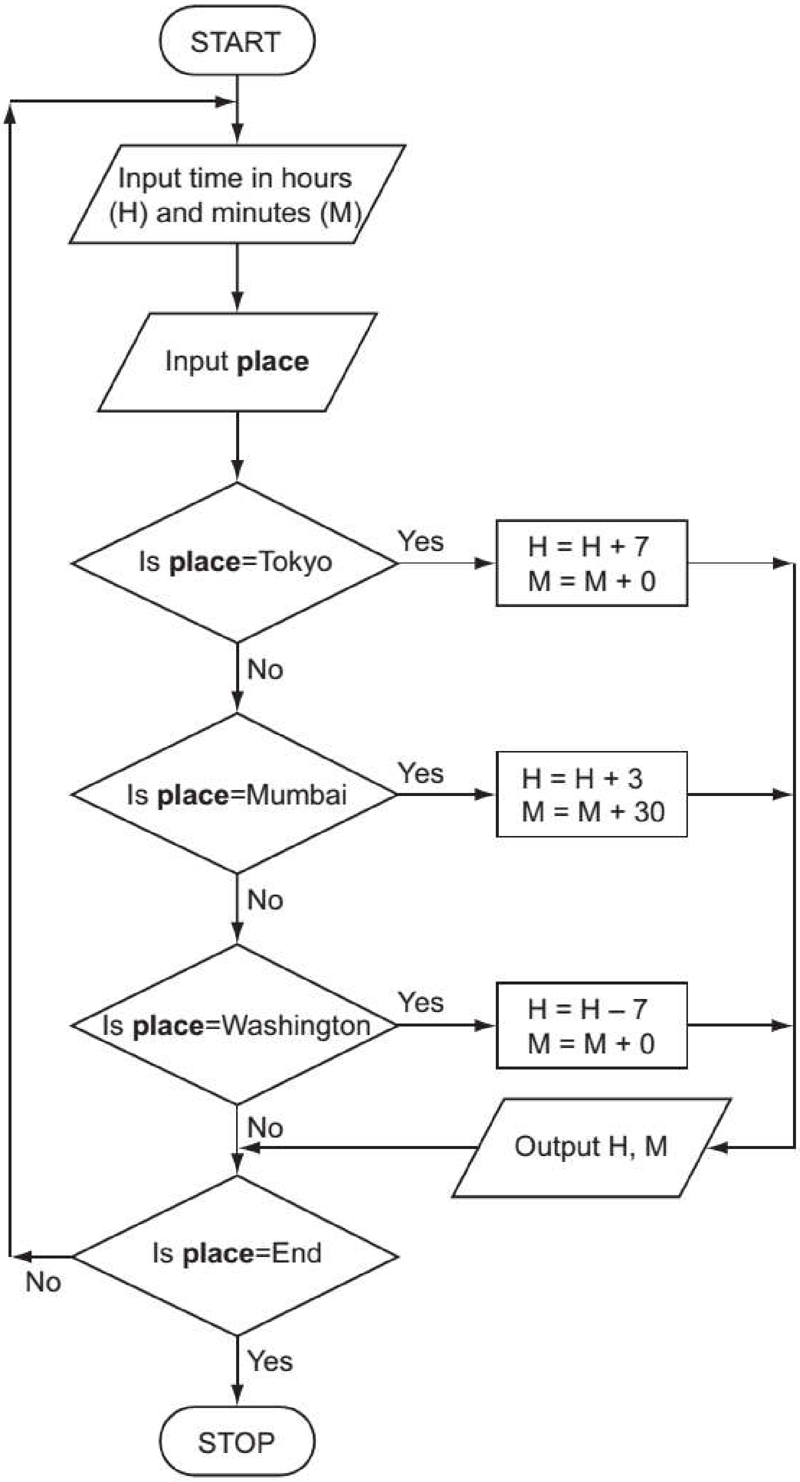


Example



Q 11.1) Majid lives in Cairo but often travels to Tokyo, Mumbai and Washington. A flow chart has been written so he can work out the local time in these three places.

Improve this flowchart



Iteration (Repetition, Loop)

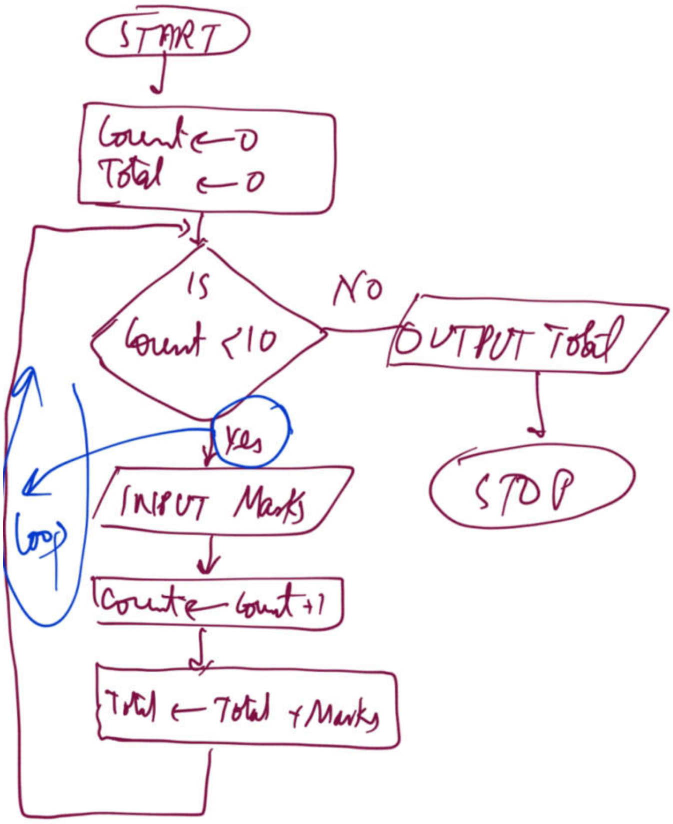
Repetition is used to execute a set of instructions multiple times.
Repetition is also referred as LOOP or ITERATION.
There are following three types of loops:

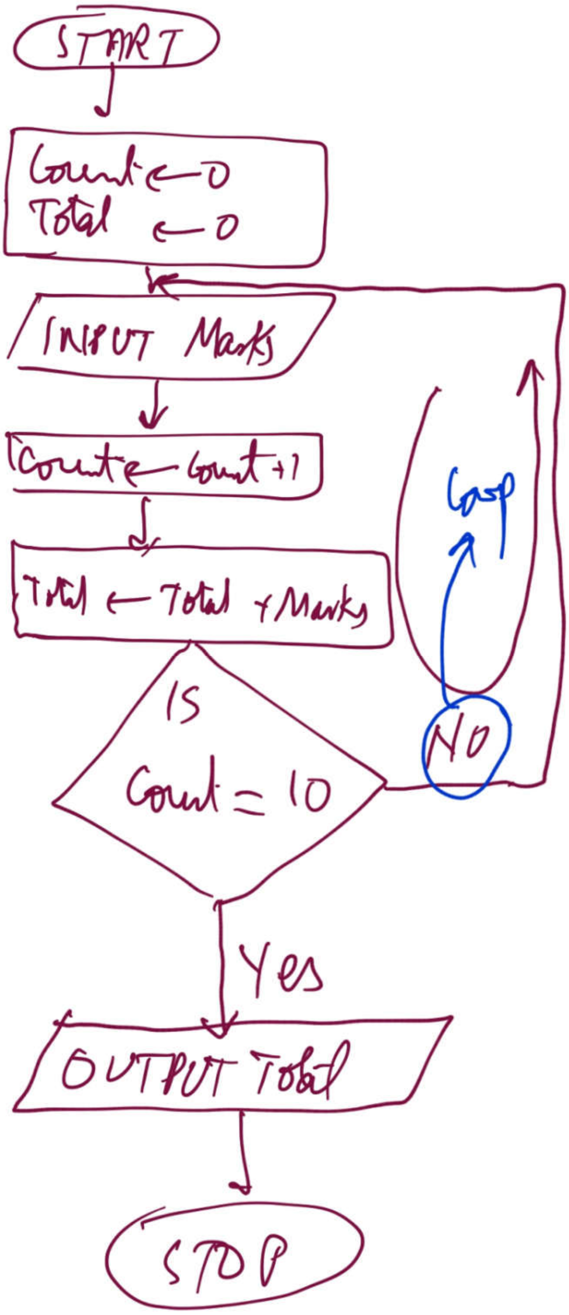
1. Count-controlled loop
2. Pre-condition loop
3. Post-condition loop

Count-controlled (FOR) loops

Count-controlled loop is used when the number of repetition is already known.

Example: to input 10 numbers and output their final total



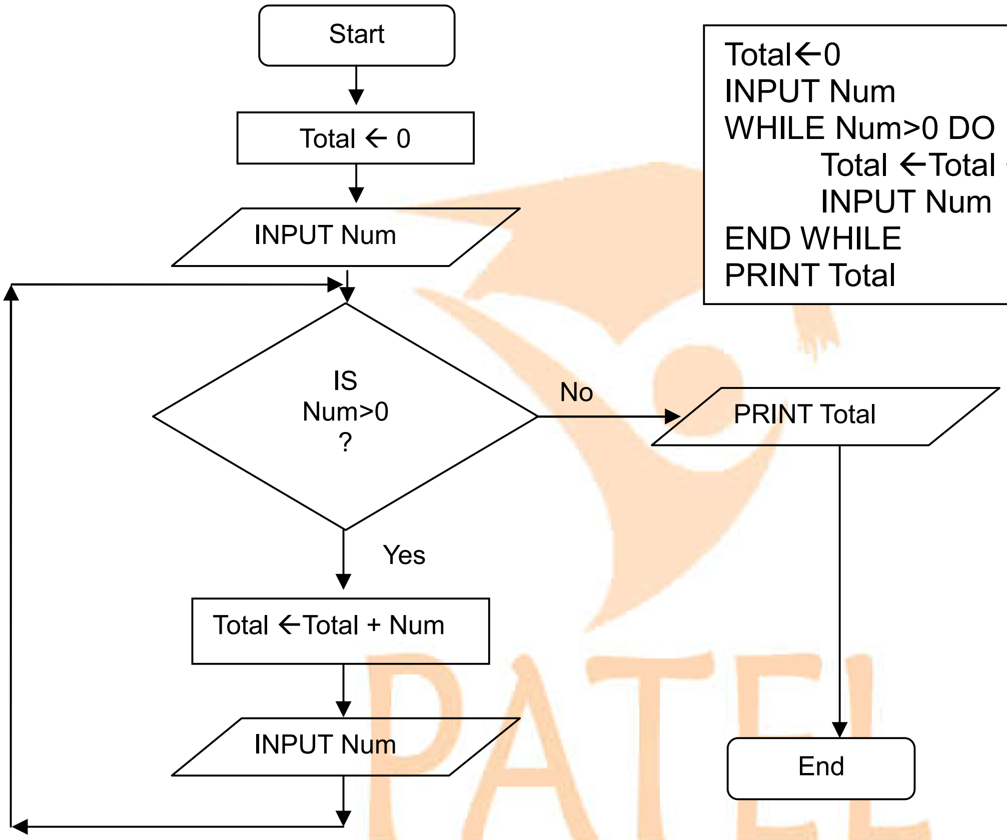


Pre-condition (WHILE) loops

A loop in which condition is given at the start of loop and which is executed only when the condition is true is called post-condition loop.

Example: To input a series of numbers, calculate total and stops if a –ve number is entered:

The condition is checked at the beginning of the loop. If condition is true loop statements are executed again and again.



```
Total←0
INPUT Num
WHILE Num>0 DO
          Total ←Total + Num
          INPUT Num
END WHILE
PRINT Total
```

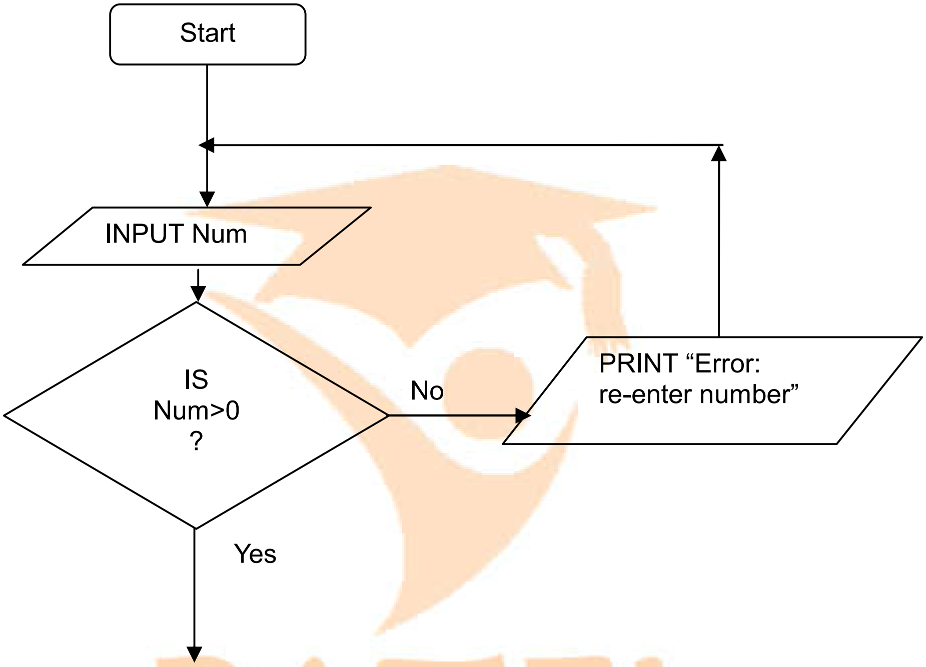
Post-condition (REPEAT UNTIL) loops

A loop in which condition is given at the end of loop and which is executed only when the condition is false is called post-condition loop.

Example: To input and validate a number and to reject it if a negative number is entered and ask to re-enter another number

The condition is checked at the end of the loop. If condition is false loop statements are executed again and again.

```
REPEAT
        INPUT Num
UNTIL Num>0
```



Example Question

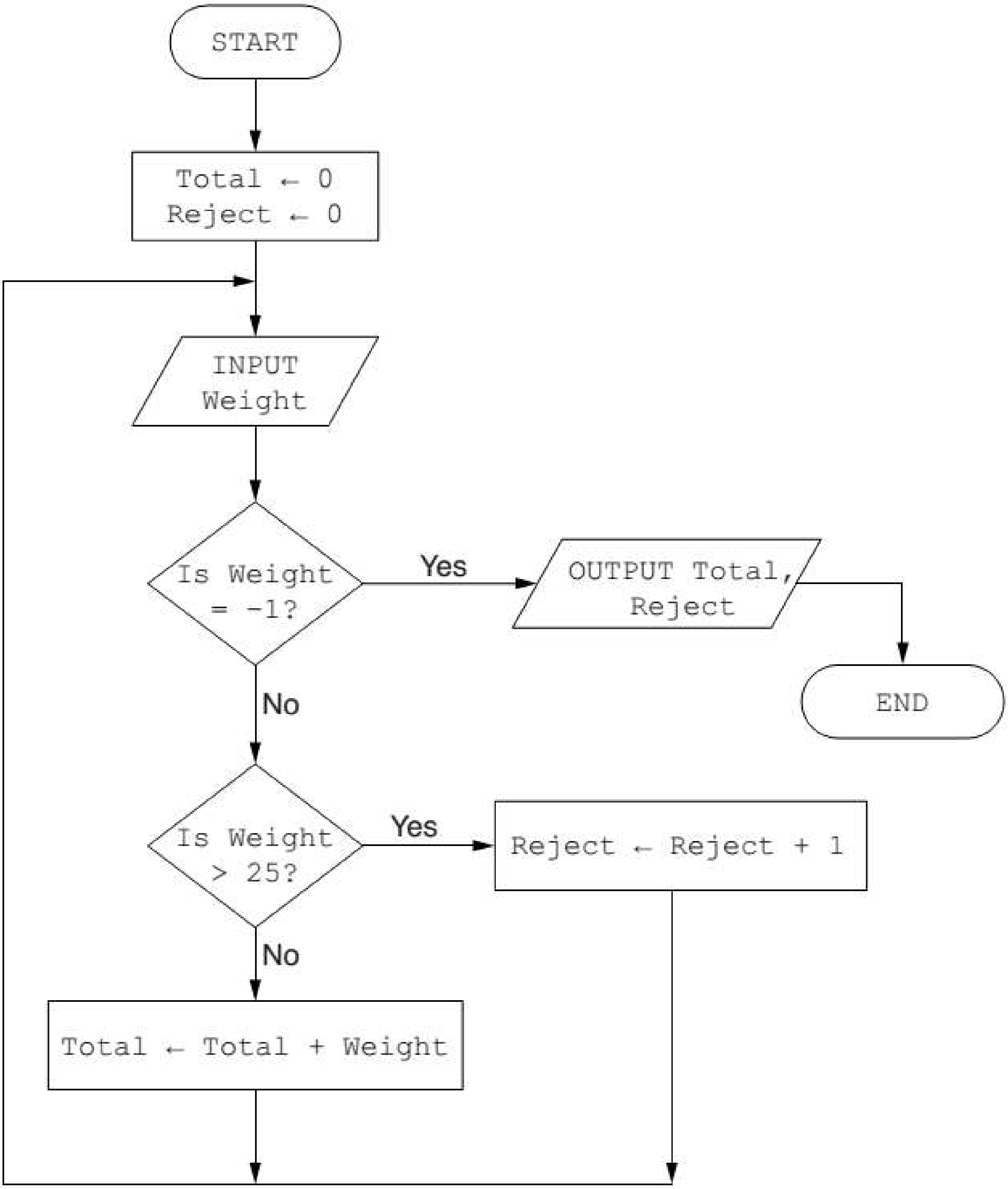
Q 11.2) Draw a flowchart that

Inputs the weight of a number of parcels in kilograms.
Validate parcel (parcels weighing more than 25 kilograms are rejected).
A value of –1 stops the input.
Outputs the total weight of the parcels accepted and number of parcels rejected.

Past Paper Question of same type in Summer 2015 P21& 23 Q 3

The flowchart below inputs the weight of a number of parcels in kilograms. Parcels weighing more than 25 kilograms are rejected. A value of –1 stops the input.

The following information is output: the total weight of the parcels accepted and number of parcels rejected.



Complete the trace table for the input data:

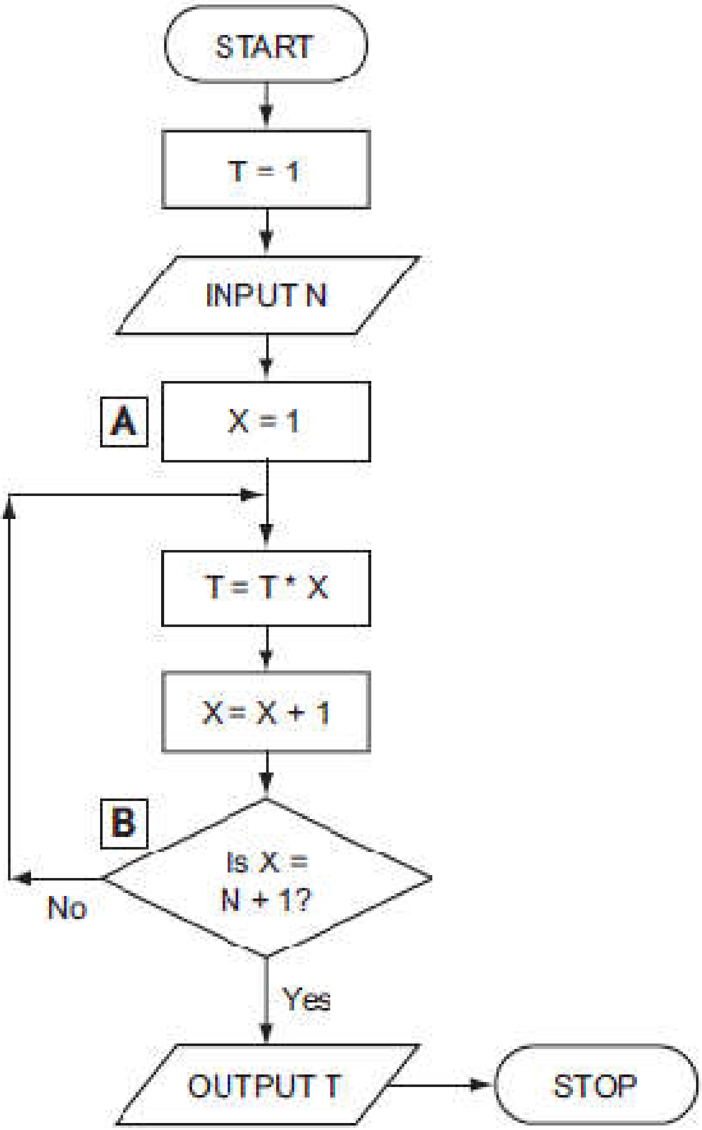
1.8, 26.0, 7.0, 11.3, 10.0, 2.5, 25.2, 5.0, 19.8, 29.3, –1 [5]

Total	Reject	Weight	OUTPUT

Examiner Report Question 3

Many candidates showed the skill of using a trace table for data entry; most candidates correctly initialized the variables, Total and Reject; some candidates did not always trace the weight checking correctly and had errors in the Total column.

Q11.3)Summer 2009
Study the flowchart very carefully.



(a) Complete the table to show what outputs you would expect for the two inputs.[2]

Input N	Output T
5	
1	

(b) Write down a possible LOOP construct for the section A to B in the flowchart using pseudo code.

……

……

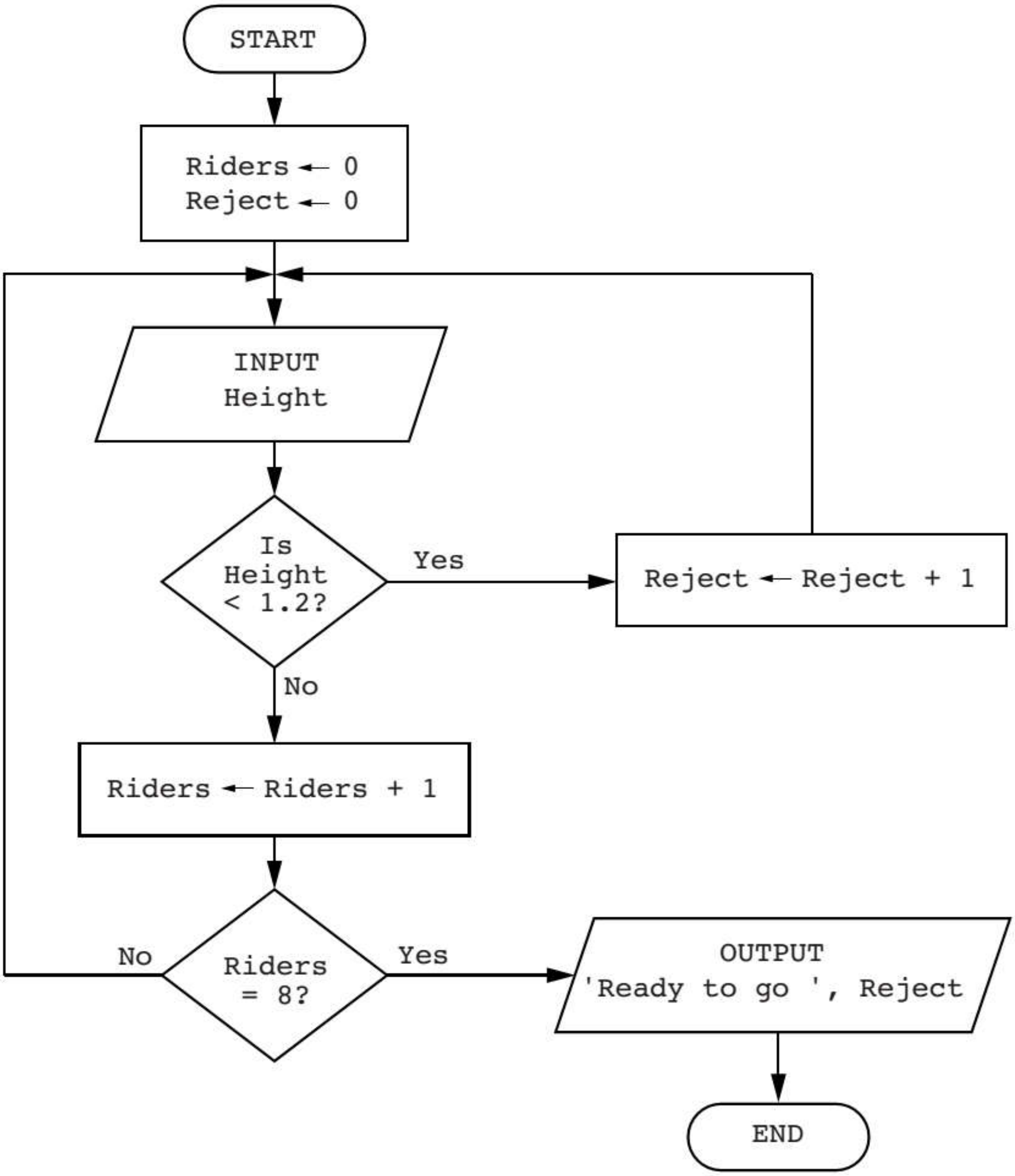
……………………………………………………………………………………….. [2]

Q 11.4) Draw an algorithm using flowchart that:

- Inputs the height of children who want to ride on a rollercoaster
- Validates height children under 1.2 metres are rejected.
- When eight children have been accepted, outputs message “Ready to go” and number of children rejected.

Past Paper Question of same type in Summer 20162210,0478 P21 &P23

4 The flowchart below inputs the height of children who want to ride on a rollercoaster. Children under 1.2 metres are rejected. The ride starts when eight children have been accepted.



Complete the trace table for the input data:

1.4, 1.3, 1.1, 1.3, 1.0, 1.5, 1.2, 1.3, 1.4, 1.3, 0.9, 1.5, 1.6, 1.0 [4]

Riders	Reject	Height	OUTPUT

Q11.5a) Write an algorithm, using flowchart only, that

- inputs three numbers
- outputs the greatest number

Q 11.5b) Write an algorithm, using flowchart only, that

- inputs three numbers
- outputs the smallest number

Q11.6) Draw a flowchart that

- Inputs a number
- Find out number is even or odd (using MOD function)
- Output Even or ODD

Q11.7) Draw a flowchart that

- Inputs 50 number
- Find out number is Integer or Real (using INT function)
- Count Integer and Odd Numbers
- Output how many were integer and odd

Q11.8) A formula for calculating the body mass index (BMI) is:

$$\text{BMI} = \frac{\text{weight in kilograms}}{(\text{height in metres}) \times (\text{height in metres})}$$

Using **Flowchart**, write an algorithm that will input weight (kg) and height (m) of students, calculate their body mass index (BMI) and output their BMI and comments on BMI.

BMI <19	Under weight
BMI < =25	Normal Weight
BMI>25	Over weight

Q11.9) A system uses 5 digit numbers with an additional sixth digit used as a check digit.
(b) Each of the six digits in the number has a digit position. [Total=6]

6	5	4	3	2	1	←Digit position
a	b	c	d	e	f	
					Check digit	

digit in position 1 is the check digit i.e. f

The validity of the check digit is found using the following calculation:

- multiply each digit by its digit position (i.e. ax6, bx5, so on)
- add together the results of the multiplications
- divide the sum by 11
- If the remainder is ZERO then the number is valid

Write an algorithm, using flowchart only, which

- inputssix-digit barcodes in the form a, b, c, d, e and f
- re-calculates the check digit for each number and checks whether the input check digit (e) is correct

Q 11.10) Summer 2013

A small shop uses barcodes which represent 5 digits. The last digit is used as a check digit.
For example:
a b c d e
0 1 2 3 4
The check digit (e) is found by:

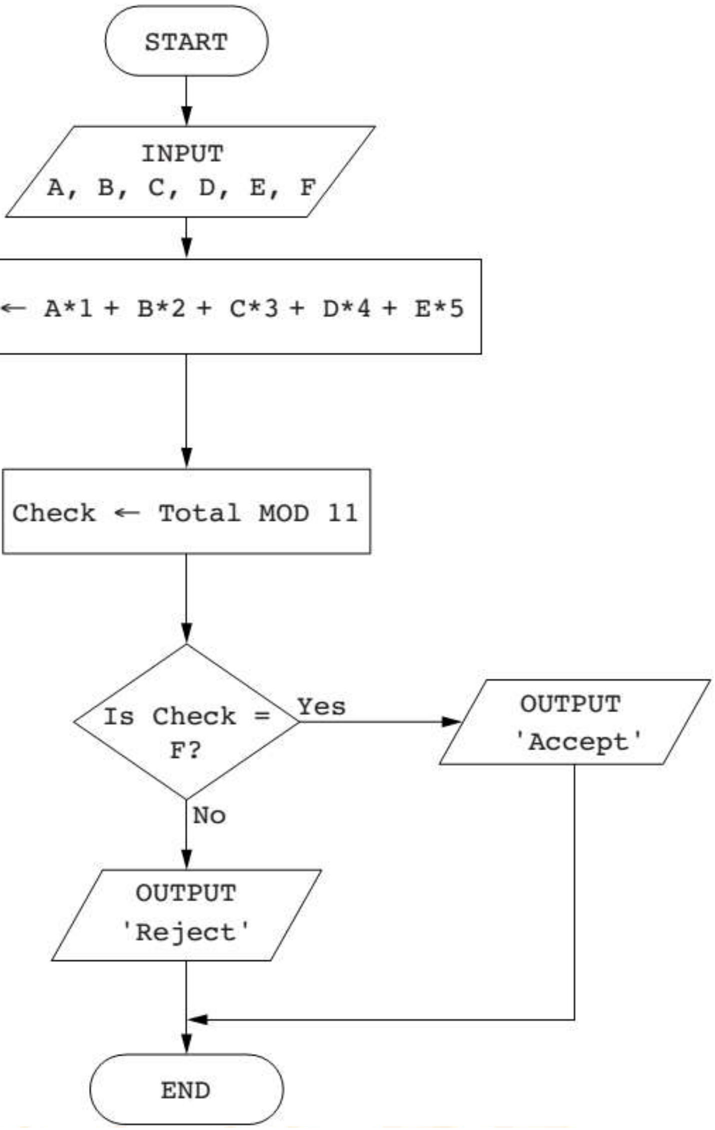
- multiplying the first and third digits (i.e. a and c) by 3
- multiplying the second and fourth digits (i.e. b and d) by 2
- adding these four results together to give a total
- dividing this total by 10
- remainder is check digit (e)

Write an algorithm, using flowchart only, which

- inputs five-digit barcodes in the form a, b, c, d, e
- re-calculates the check digit for each number and checks whether the input check digit (e) is correct

Past paper Question of same type in Summer 2015 P22

3 (a) The flowchart below inputs six single digit numbers. The predefined function `MOD` gives the value of the remainder, for example, `Y ← 10 MOD 3` gives the value `Y = 1`



Complete a trace table for each of the two sets of input data.
Set 1 5, 2, 4, 3, 1, 5
Set 2 3, 2, 1, 0, 7, 3

Trace table set 1: 5, 2, 4, 3, 1, 5

A	B	C	D	E	F	Total	Check	Output

Trace table set 2: 3, 2, 1, 0, 7, 3

A	B	C	D	E	F	Total	Check	Output

Q11.11) Draw a flowchart that

- Inputs a series of numbers
- Calculates their total
- Stops input if a negative number is entered
- Output total.

Q 11.12) Draw a flowchart that

- Inputs temperature for a week (7 days)
- Outputs highest and lowest temperature

Q 11.13) Draw a flowchart that

- Inputs marks of a class of 30 students
- Outputs how many students are pass and how many are fail

Q 11.14) Draw a flowchart that

- Inputs per litre price of 5 different brands of milk
- Outputs how average price per litre

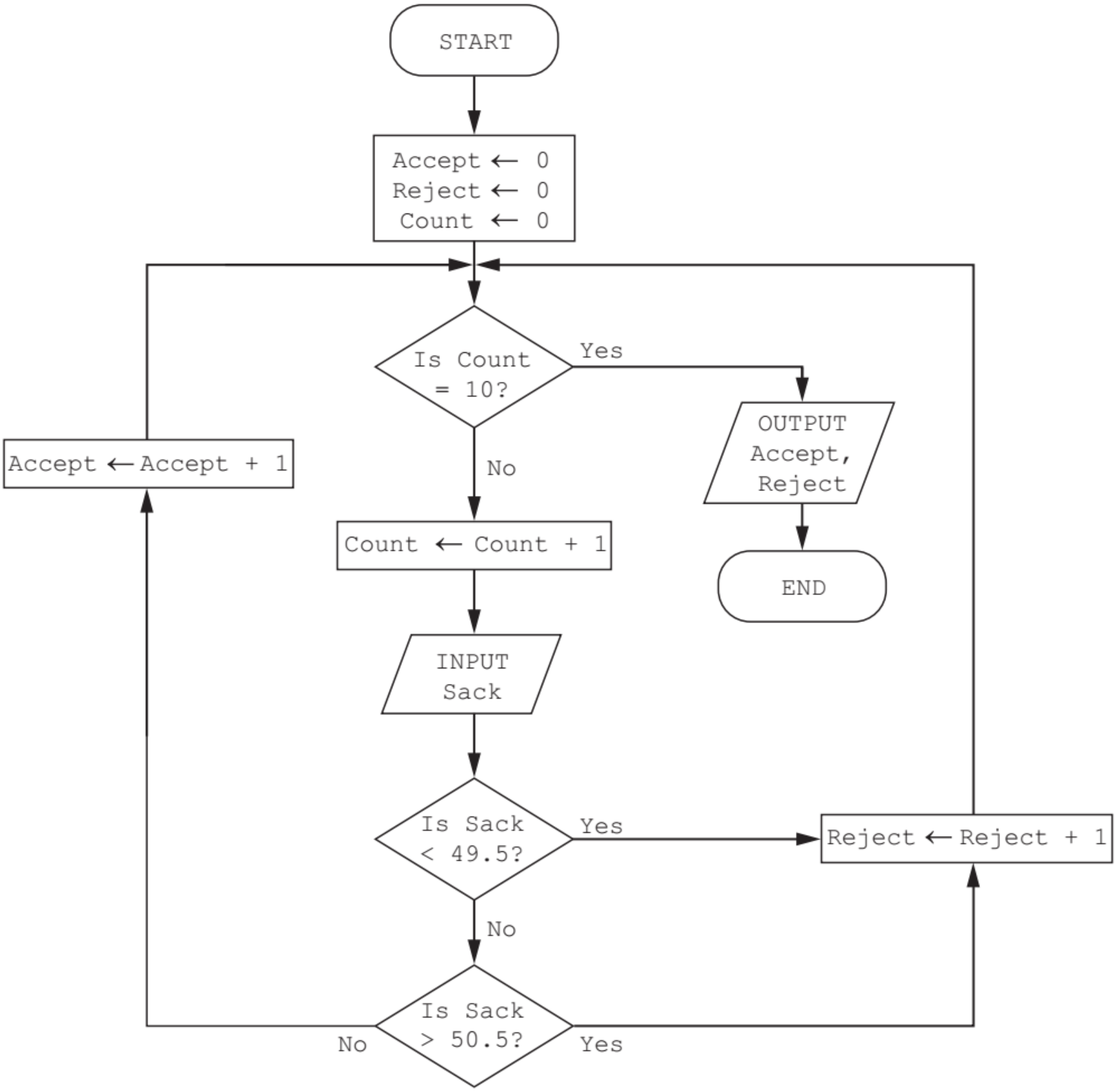
Q 11.15) Draw a flowchart that

Inputs a batch of 10 rice sacks for weight

- Validates sacks (sacks should weigh 50 kilograms each. Sacks weighing over 50.5 kilograms or less than 49.5 kilograms are rejected.)
- Outputs number of sacks accepted and the number of sacks rejected.

Past paper flowchart for same type of question in Winter 2017 P22 Q5

(a) This flowchart checks a batch of 10 rice sacks for weight. Sacks should weigh 50 kilograms each. Sacks weighing over 50.5 kilograms or less than 49.5 kilograms are rejected. The number of sacks accepted and the number of sacks rejected is output.



Complete the trace table for the input data:

50.4, 50.3, 49.1, 50.3, 50.0, 49.5, 50.2, 50.3, 50.5, 50.6 [5]

Accept	Reject	Count	Sack	OUTPUT

(b) The size of the batch has increased to 50 sacks. It has been decided to only reject sacks thatare underweight.

State the changes that need to be made to the flowchart.

....

....

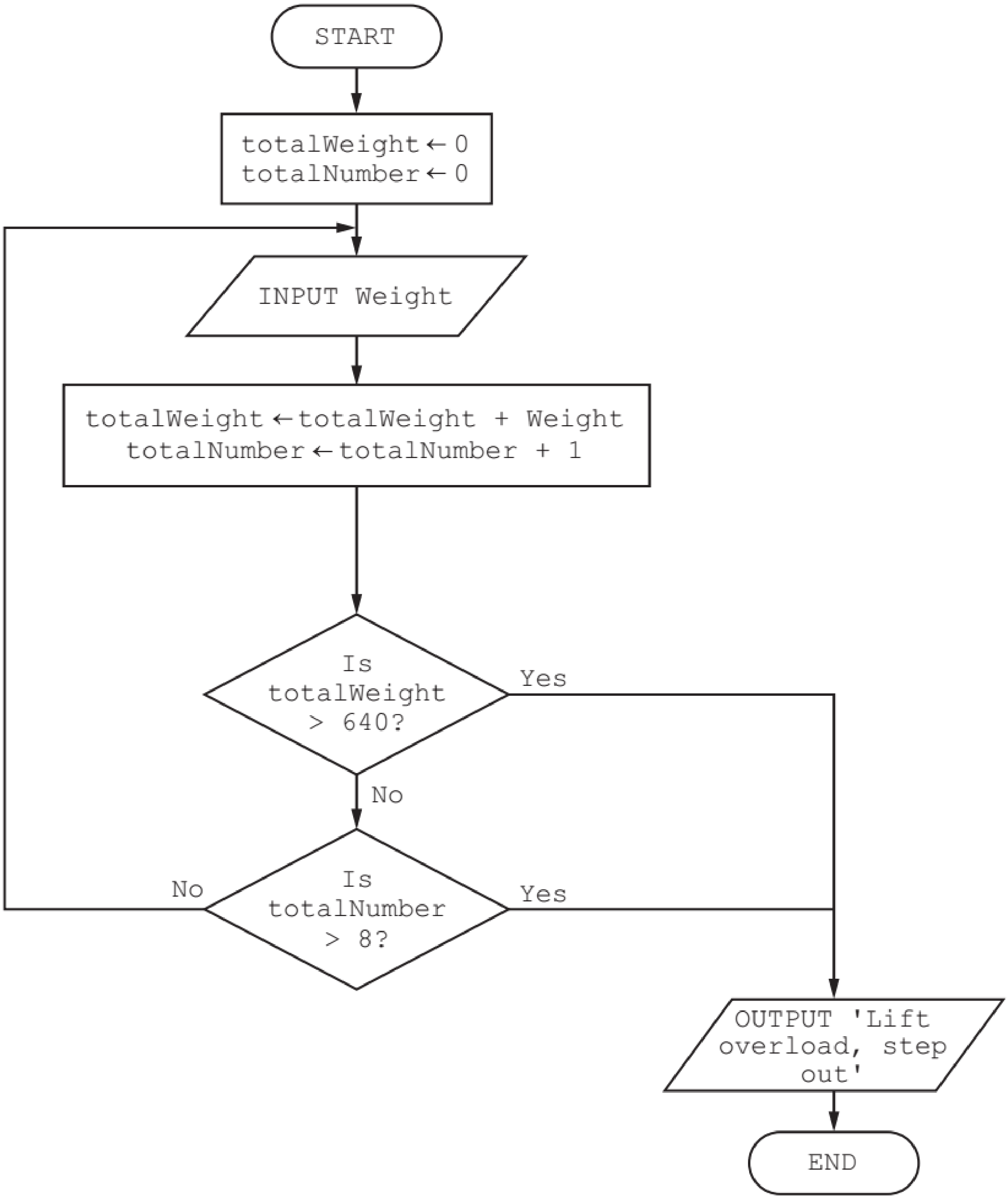
....

....[2]

Q11.16) Draw a flowchart that
inputs the weight in kilograms of a passenger stepping into a lift.
The lift can take a maximum of eight passengers or a maximum weight of 640 kilograms.

March 2018 P22 (India)

3 This flowchart inputs the weight in kilograms of a passenger stepping into a lift. The lift can take a maximum of eight passengers or a maximum weight of 640 kilograms.



Complete the trace table for the passenger input data: [4]
50, 70, 65, 100, 95, 50, 55, 85, 70, 75

Weight	TotalWeight	TotalNumber	OUTPUT

Comments on Question 3

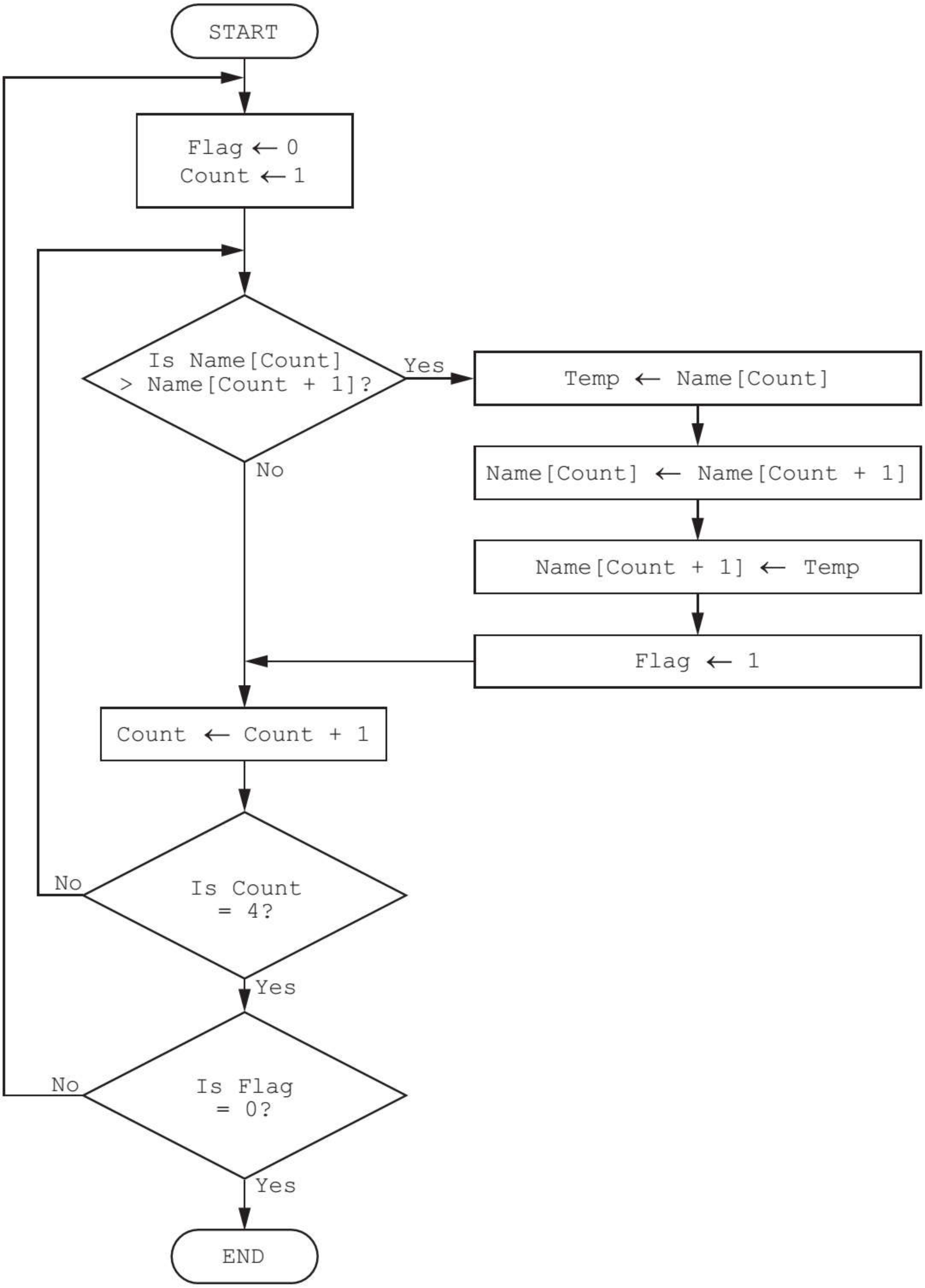
Most candidates showed the skill of using a trace table. Some candidates provided a 'rough answer' in pencil and a final answer in ink; this is not recommended as extra values can be seen in the trace table. Candidates found the output the most challenging column to complete correctly and common errors seen were to incorrectly include quotation marks around the output or repeats of the output.

Q 11.17) Draw a flowchart that

- Inputs name of 10 students in a class and store them in one dimension array
- Display list of names of students

Past paper flowchart for same type of question in Winter 2017 P21 Q5

The flowchart below represents a program routine.



(a) The array used in the flowchart contains the following data:

Name[1]	Name[2]	Name[3]	Name[4]
Jamal	Amir	Eve	Tara

Complete the trace table using the data given in the array. [5]

Flag	Count	Name[1]	Name[2]	Name[3]	Name[4]	Temp
		Jamal	Amir	Eve	Tara	

(b) Describe what the algorithm represented by the flowchart is doing.

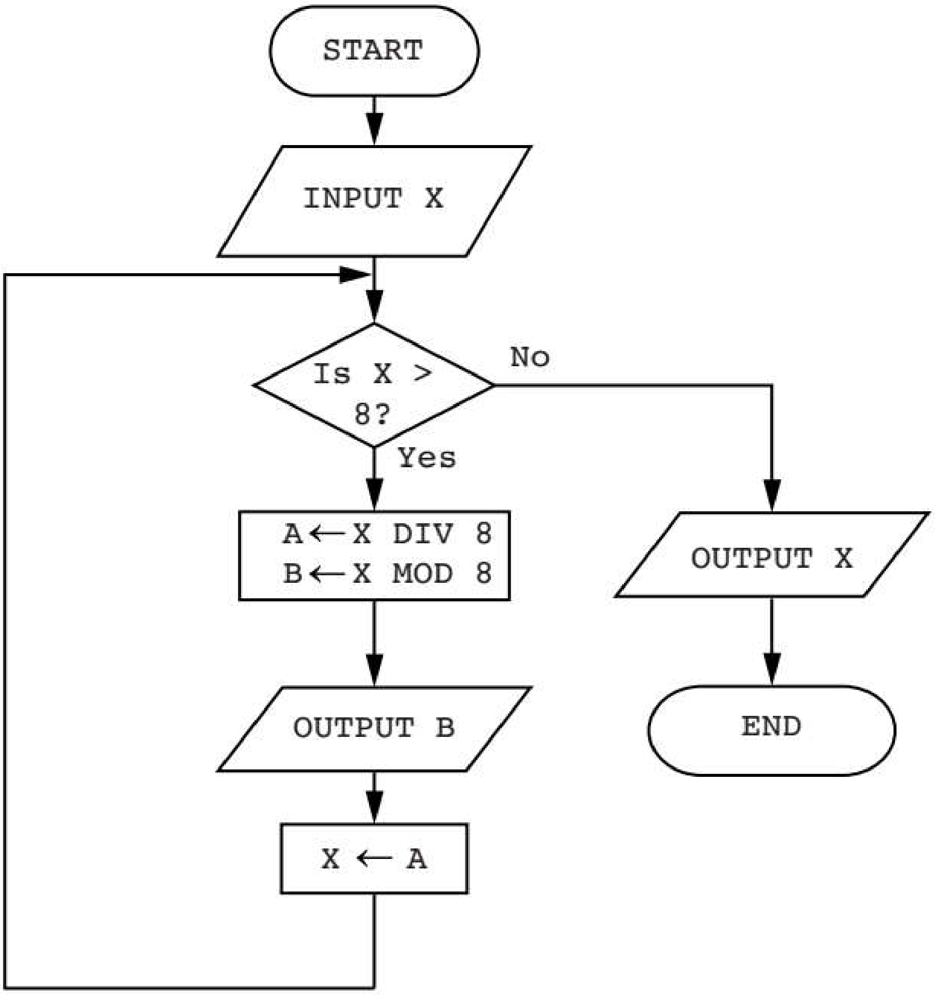
....

....[2]

Q 11.18)Winter 2016 P22 Q 3

Following flowchart is used to convert a denary number into octal (base 8)

The flowchart below inputs an integer. The predefined function DIV gives the value of the division, for example Z ←11 DIV 3 gives the value Z = 3. The predefined function MOD gives the value of the remainder, for example Z ←11 MOD 3 gives the value Z = 2.



Complete a trace table for each of the two input values **33** and **75**. [4]

Trace table for input value **33**

X	A	B	OUTPUT

Trace table for input value **75**

X	A	B	OUTPUT

Q 11.19) Draw a flowchart to convert denary number into binary

Test your flowchart by completing following trace table to convert denary number 20 into binary [4]

X	A	B	OUTPUT

Q 11.20)Winter 2015 P23 Q 3

(a) This pseudo code inputs an integer. The predefined function DIV gives the value of the division, e.g. Y 10 DIV 3 gives the value Y = 3. The predefined function MOD gives the value of the remainder, e.g. Y 10 MOD 3 gives the value Y = 1.gives the value Y = 1.

```
INPUT X
WHILE X > 15
  DO
  T1 ← X DIV 16
  T2 ← X MOD 16
  CASE T2 OF
    10:OUTPUT A
    11:OUTPUT B
    12:OUTPUT C
    13:OUTPUT D
    14:OUTPUT E
    15:OUTPUT F
    OTHERWISE OUTPUT T2
  ENDCASE
  X ← T1
ENDWHILE
CASE X OF
  10:OUTPUT A
  11:OUTPUT B
  12:OUTPUT C
  13:OUTPUT D
  14:OUTPUT E
  15:OUTPUT F
  OTHERWISE OUTPUT X
ENDCASE
```

Complete a trace table for each of the **two** input values 37 and 191.

Trace table for input value 37

X	T1	T2	OUTPUT

Trace table for input value 191

X	T1	T2	OUTPUT

(b) State the purpose of the pseudo code in **part (a)**.

....

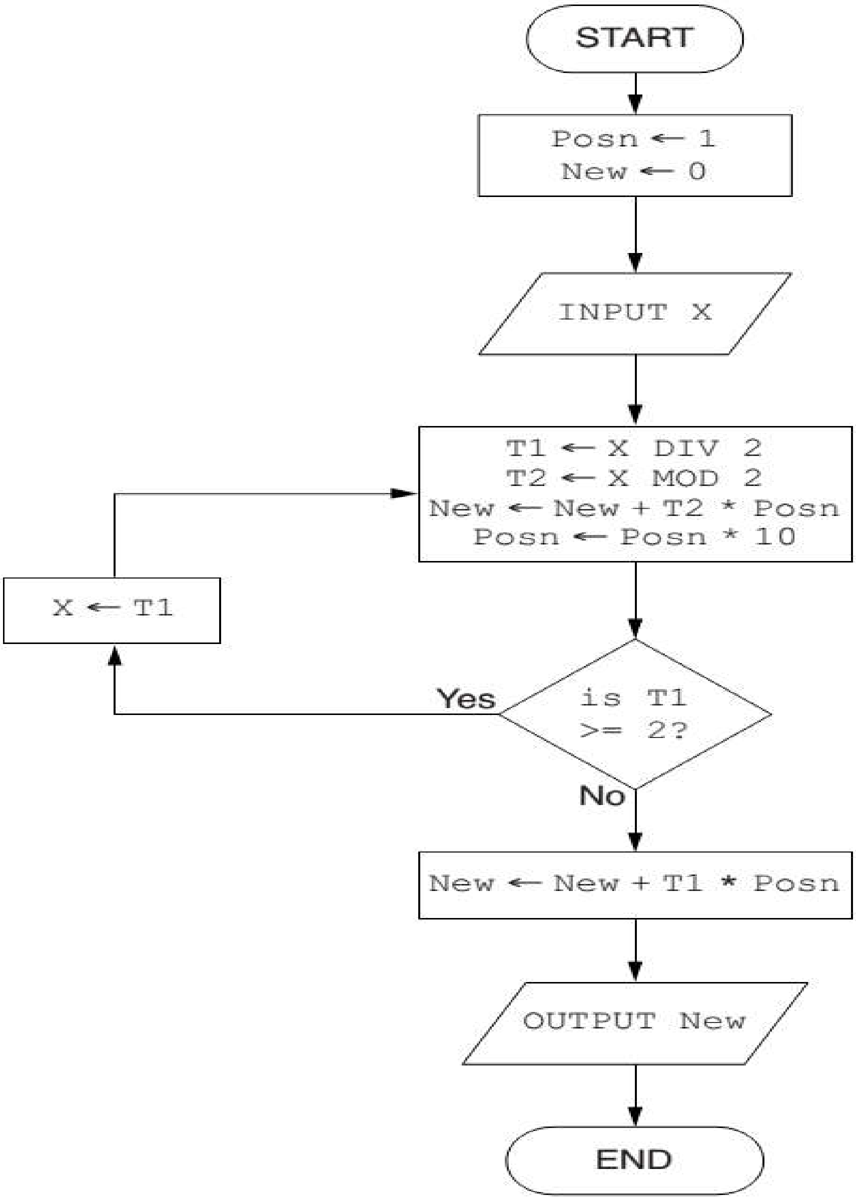
....[2]

Draw flowchart for the above pseudo code

Past paper question of same type in Winter 2015 P21 & 22

3 (a) The flowchart inputs an integer. The predefined function DIV gives the integer result of the division, e.g. Y 10 DIV 3 gives the value Y = 3. The predefined function MOD gives the value of the remainder, e.g. Y 10 MOD 3 gives the value Y = 1.

Trace table for input value 5



X	Posn	New	T1	T2	OUTPUT

Trace table for input value 12

X	Posn	New	T1	T2	OUTPUT

(b) State the purpose of the flowchart in **part (a)**.

....[1]

Examiners’Comments Question 3

(a) Candidates with stronger responses throughout showed the skill of using a trace table; some candidates correctly updated the variables, T1 and T2, only.

(b) Candidates with stronger responses throughout correctly identified the purpose of the flowchart as converting a denary number to binary. A few candidates incorrectly quoted the answer of a check digit from a previous examination paper.

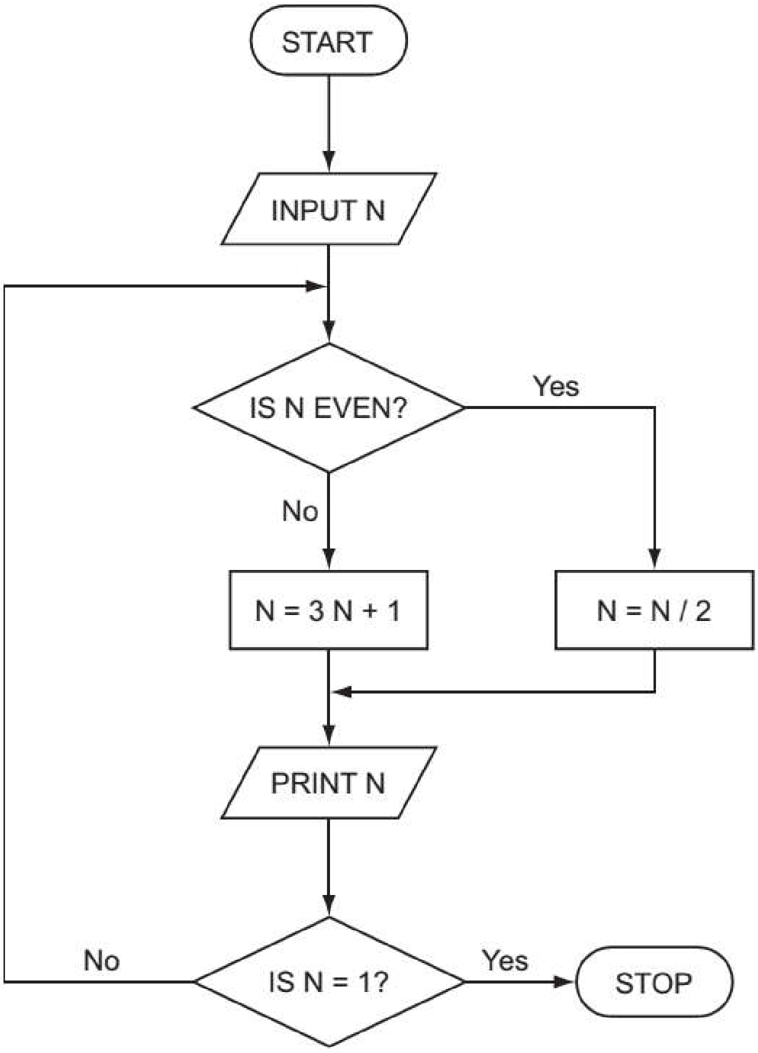
Q 11.20) Following flowchart is used to count digits in a number

```
DECLARE Count: Integer
DECLARE Num, x: Real
Count ← 0
INPUT Num
x ← PIN
REPEAT
    x ← x/10
    Count ← Count + 1
UNTIL x < 1
OUTPUT Count
```

Draw flowchart for the above pseudo code

Finding Output from flowchart

Q 11.21)Summer 2006



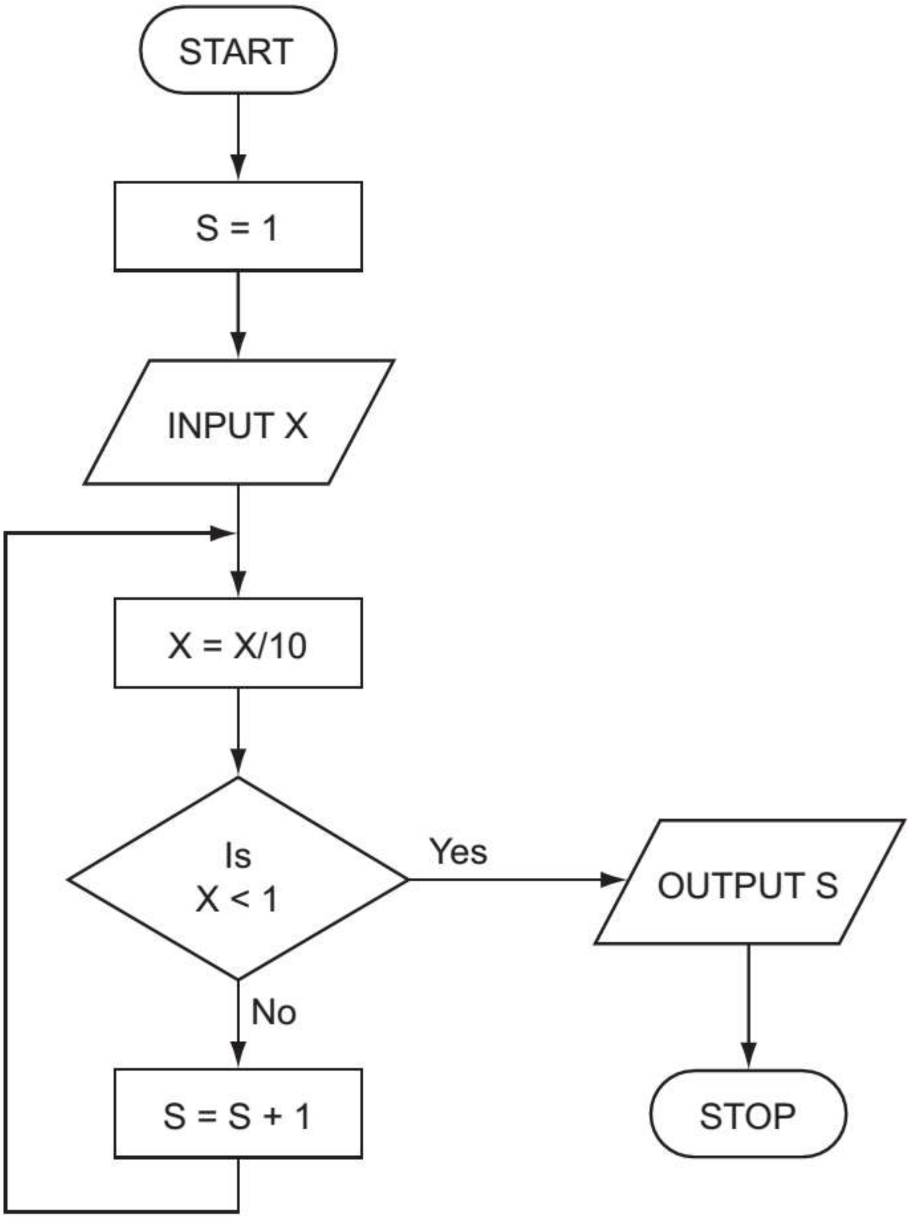
Trace the flow chart using the numbers 2 and 3. Write down each of the values of N in the order that they are printed out.

(a) 2 ....[1]

(b) 3 ....[2]

Q11.22)Summer 2007

Study the following flowchart very carefully.



(a) Complete the following table showing the expected output from the flowchart for the three sets of input data: [3]

INPUT X	OUTPUT S
48	
9170	
- 800	

(b) Input data needs to go through a validation process.

(i) Explain the term validation.

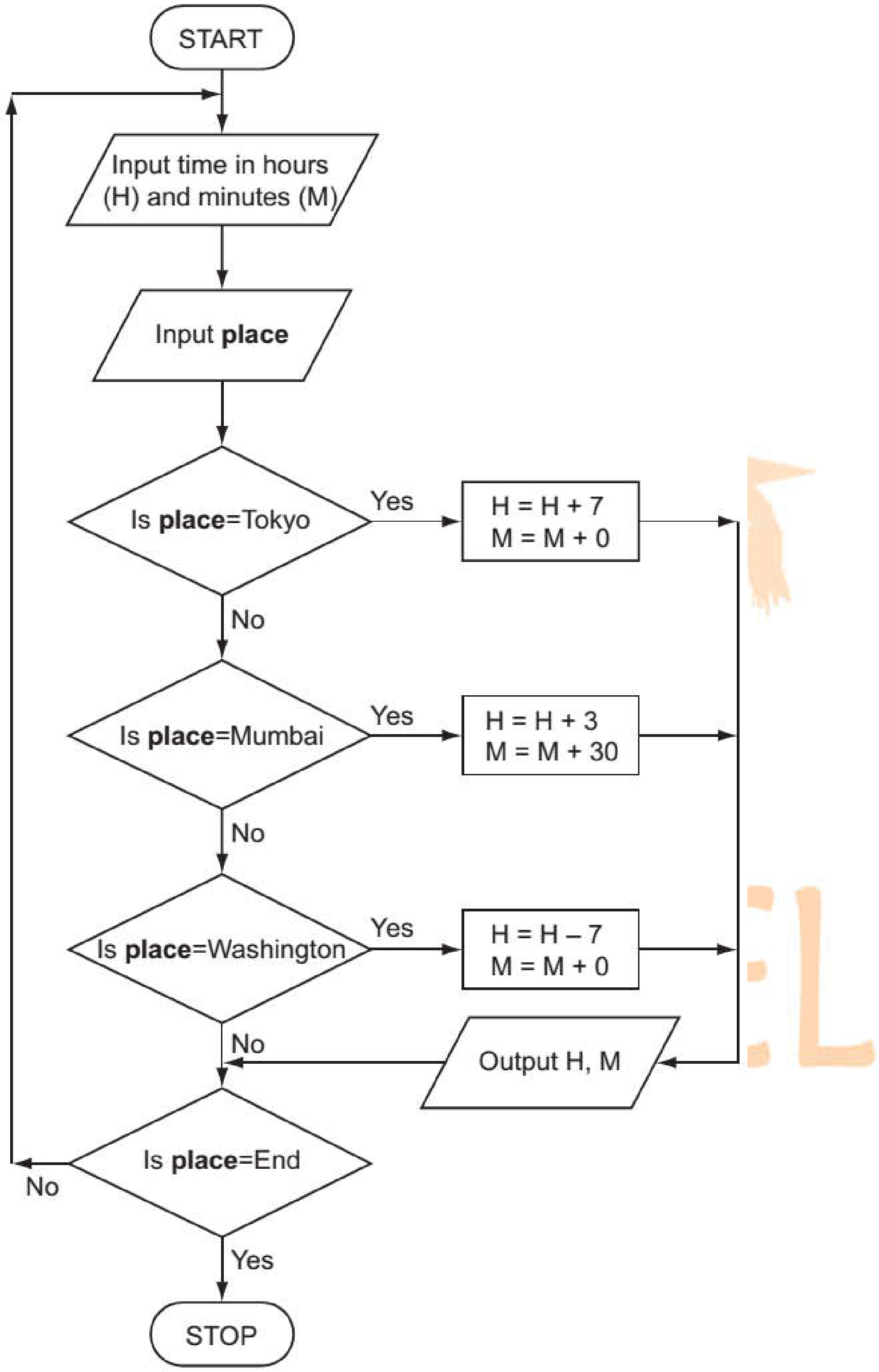
……………………………………………………………………………………………

(c) (ii) Describe one type of validation check

………………………………………………………………………………………… [2]

Q 11.23) Winter 2007

Majid lives in Cairo but often travels to Tokyo, Mumbai and Washington. A flow chart has been written so he can work out the local time in these three places.



(a) What output would be produced from the following input? [2]

Input			Output	
place	hours (H)	minutes (M)	H	M
Tokyo	11	15		
Mumbai	15	10		

(b) What problem would occur if place = Mumbai and H = 15 and M = 30?

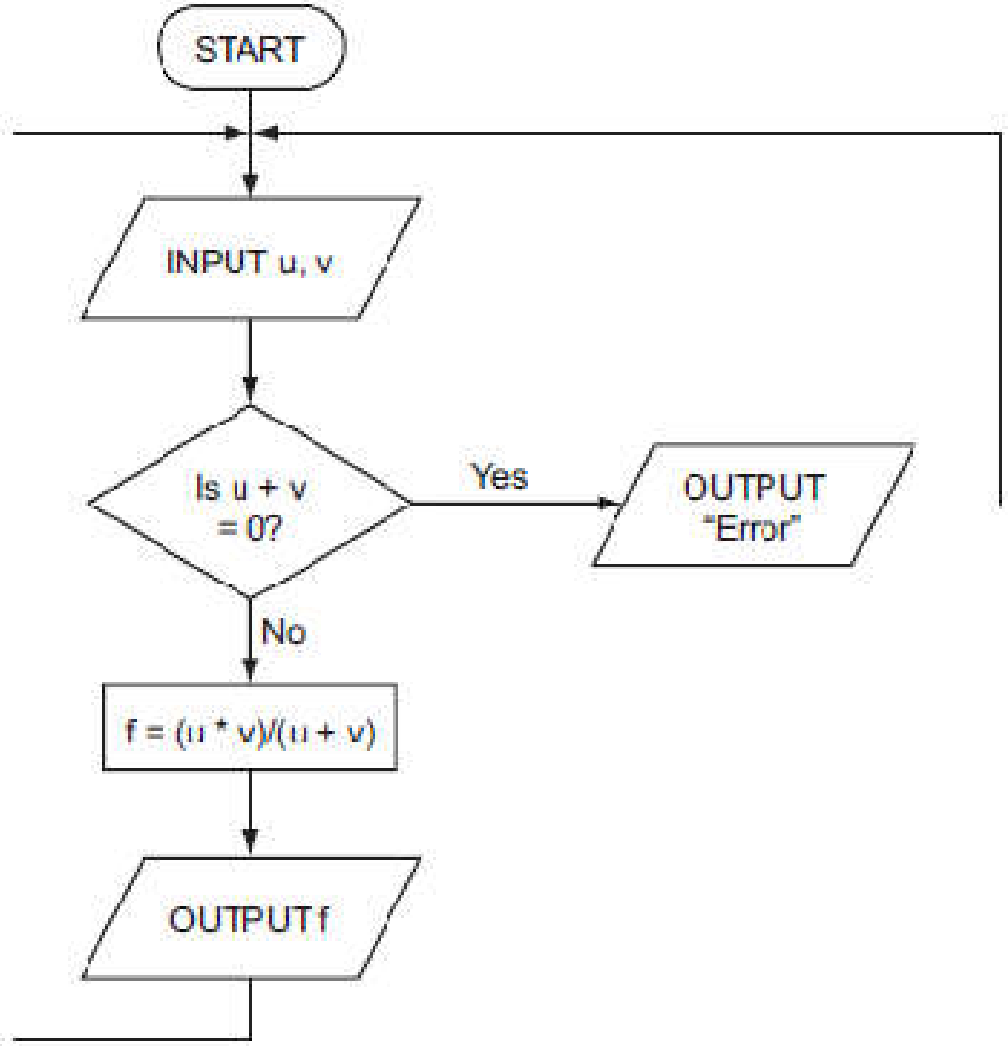
.... [1]

(c) What problem would occur if place = Washington and H = 4 and M = 0?

.... [1]

Q 11.24) Summer 2008

The following flowchart inputs two numbers, carries out a calculation and then outputs the result.



(a) Complete the following table for the three sets of input data. [3]

INPUT		OUTPUT
U	V	
5	5	
6	-6	
12	4	

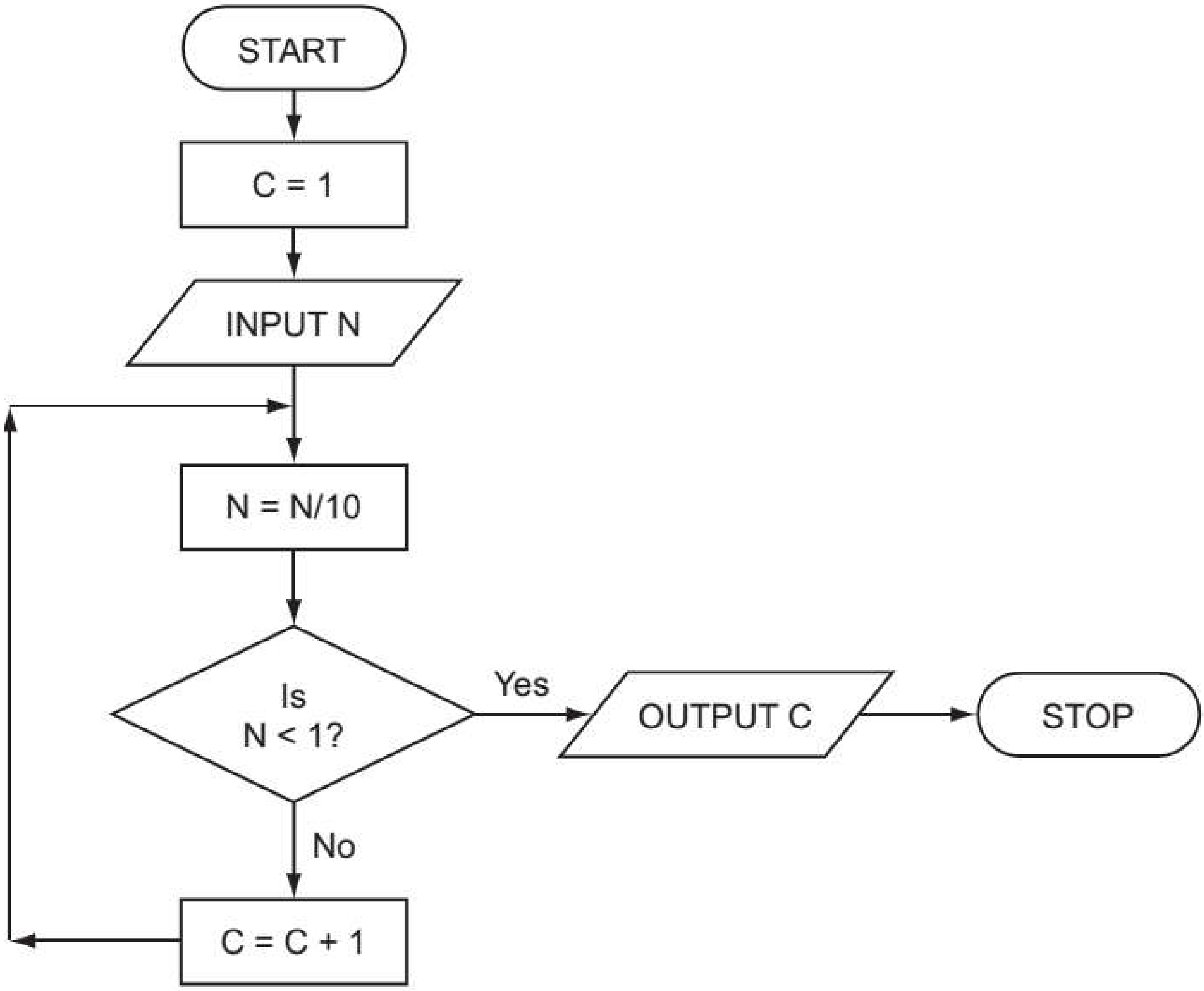
(b) The above algorithm has been placed in a library of routines. Give one advantage of doing this.

....

....[1]

Q 11.25)Winter 2009. P11

Study the flowchart.

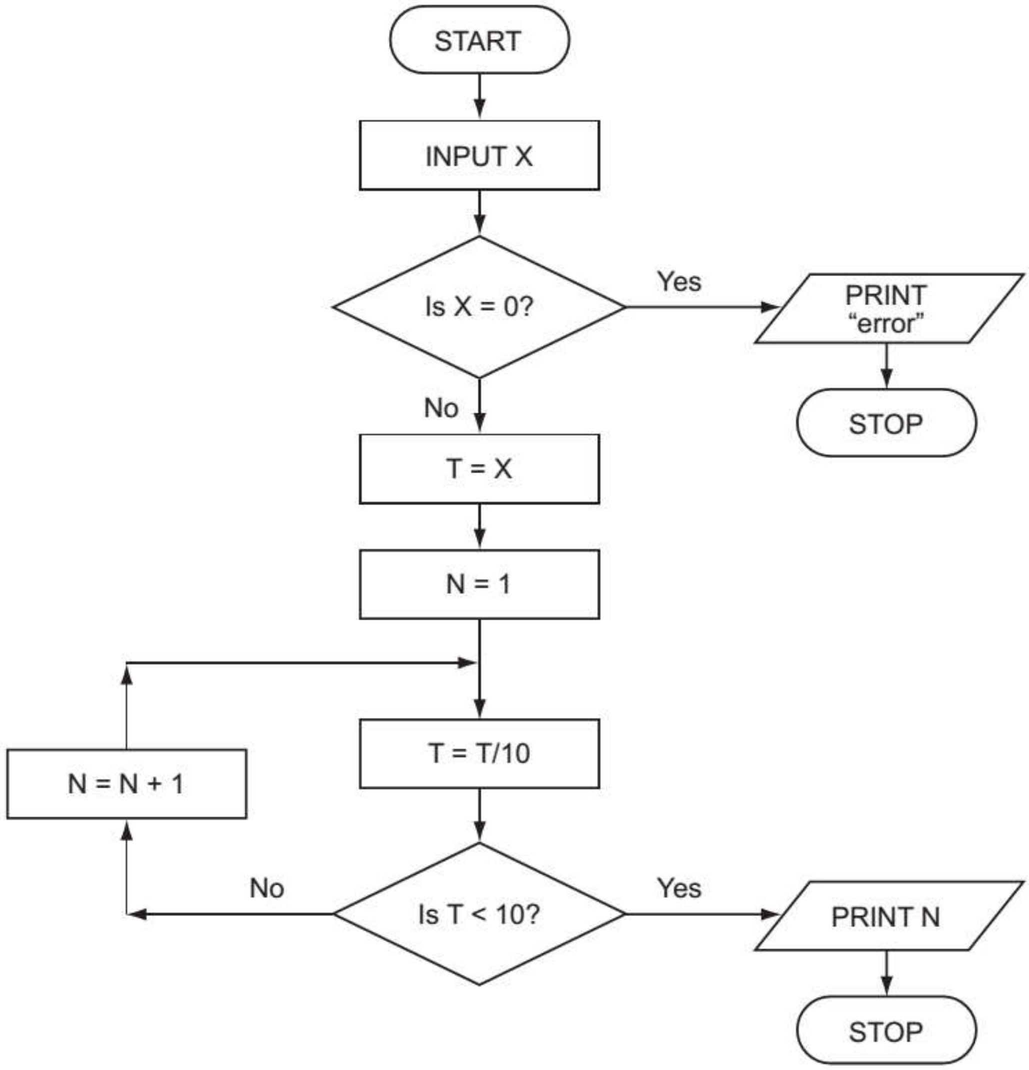


Complete the table to show what outputs you would expect for the three inputs. [3]

INPUT N	OUTPUT C
55	
2100	
1	

Q 11.26) Summer 2010 P12

Study the following flowchart very carefully:



What output would you expect if the following data was input into the flowchart? [3]

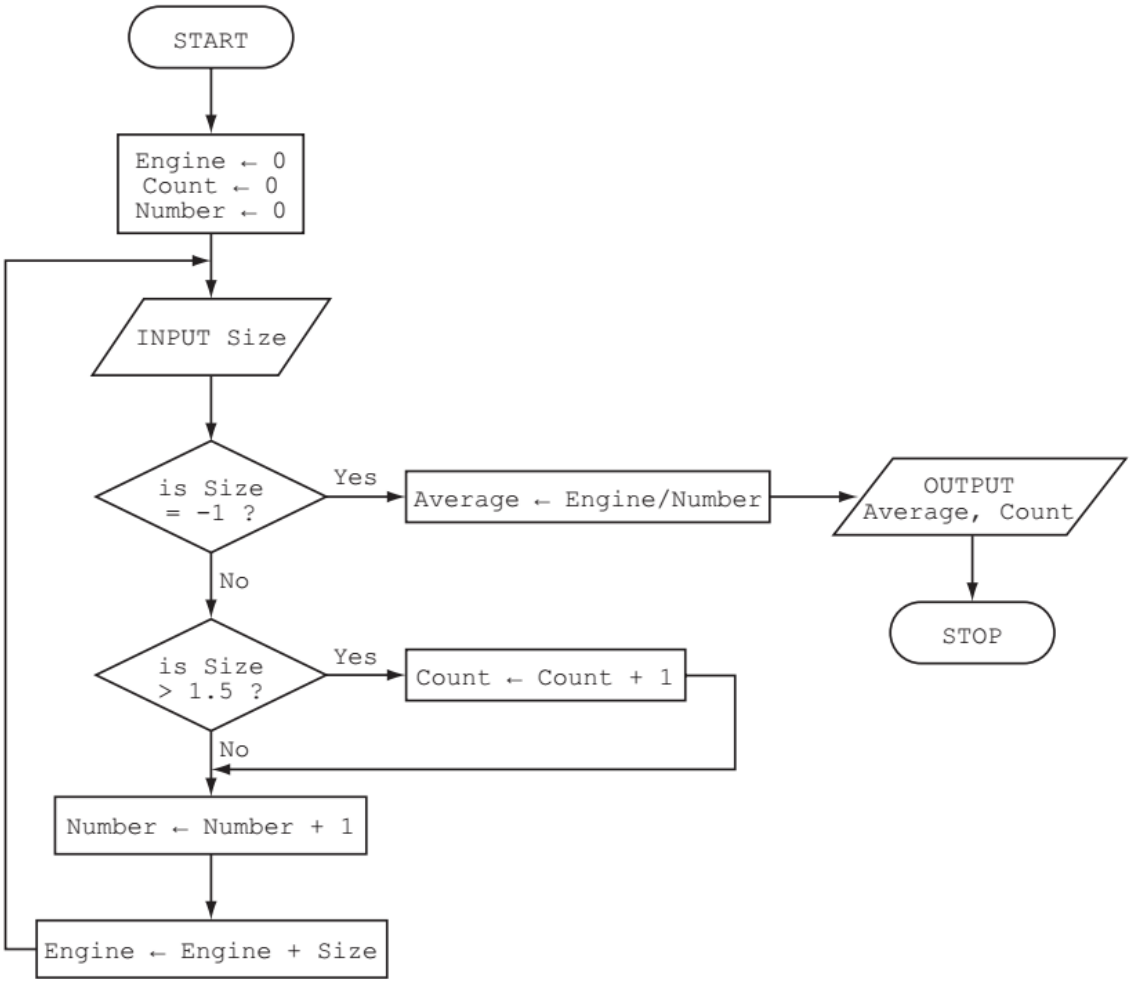
X	OUTPUT
-150	
540	
0	

`

Past paper questions from Flow Chart

Q 11.27 Specimen paper 20162210,0478 P2

3 The flowchart inputs the size of a number of car engines; a value of –1 stops the input.
This information is output: average engine size and number of engines with size > 1.5

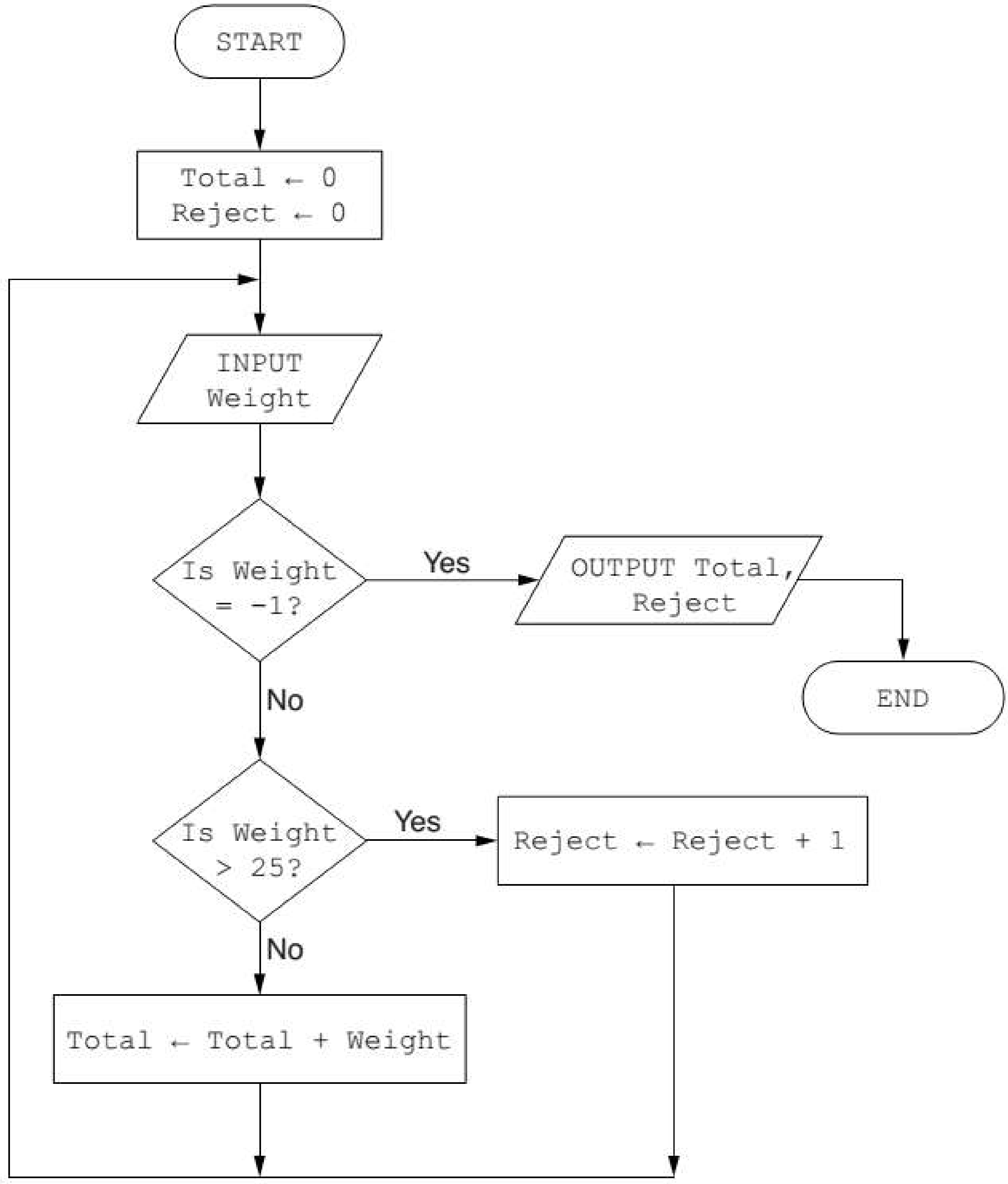


Complete the trace table for the input data.: 1.8, 2.0, 1.0, 1.3, 1.0, 2.5, 2.0, 1.3, 1.8, 1.3, –1[6]

Engine	Count	Number	Size	Average	OUTPUT

Q 11.28 Summer 2015 P21& 23

3 The flowchart below inputs the weight of a number of parcels in kilograms. Parcels weighing more than 25 kilograms are rejected. A value of –1 stops the input.
The following information is output: the total weight of the parcels accepted and number of parcels rejected.



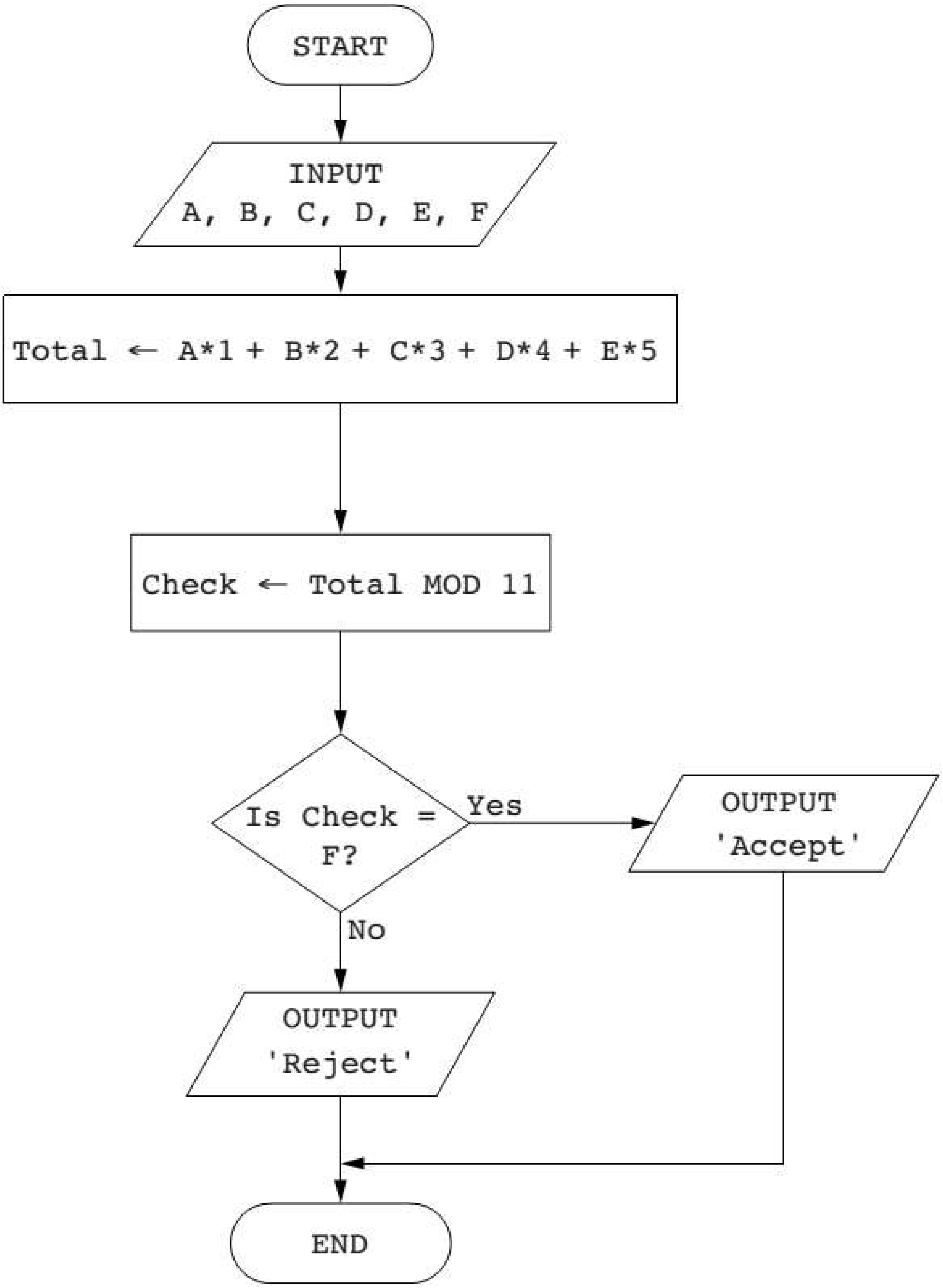
Complete the trace table for the input data:

1.8, 26.0, 7.0, 11.3, 10.0, 2.5, 25.2, 5.0, 19.8, 29.3, –1 [5]

Total	Reject	Weight	OUTPUT

Q 11.29 Summer 2015 P22

3 (a) The flowchart below inputs six single digit numbers. The predefined function `MOD` gives the value of the remainder, for example, `Y` ← `10 MOD 3` gives the value `Y = 1`



Complete a trace table for each of the two sets of input data.
Set 1 5, 2, 4, 3, 1, 5
Set 2 3, 2, 1, 0, 7, 3
Trace table set 1: 5, 2, 4, 3, 1, 5

A	B	C	D	E	F	Total	Check	Output

Trace table set 2: 3, 2, 1, 0, 7, 3

A	B	C	D	E	F	Total	Check	Output

(b) State the purpose of the flowchart in **part (a)**.
....
....[1]

(c) Identify a problem with this flowchart and explain how to correct it.
Problem ....
....
Solution ....
....
....
....[3].

<u>Examiner's comments on Question 3</u>

(a) Many candidates showed the skill of using a trace table for data entry, better candidates correctly updated the variables, Total, Check and Output.

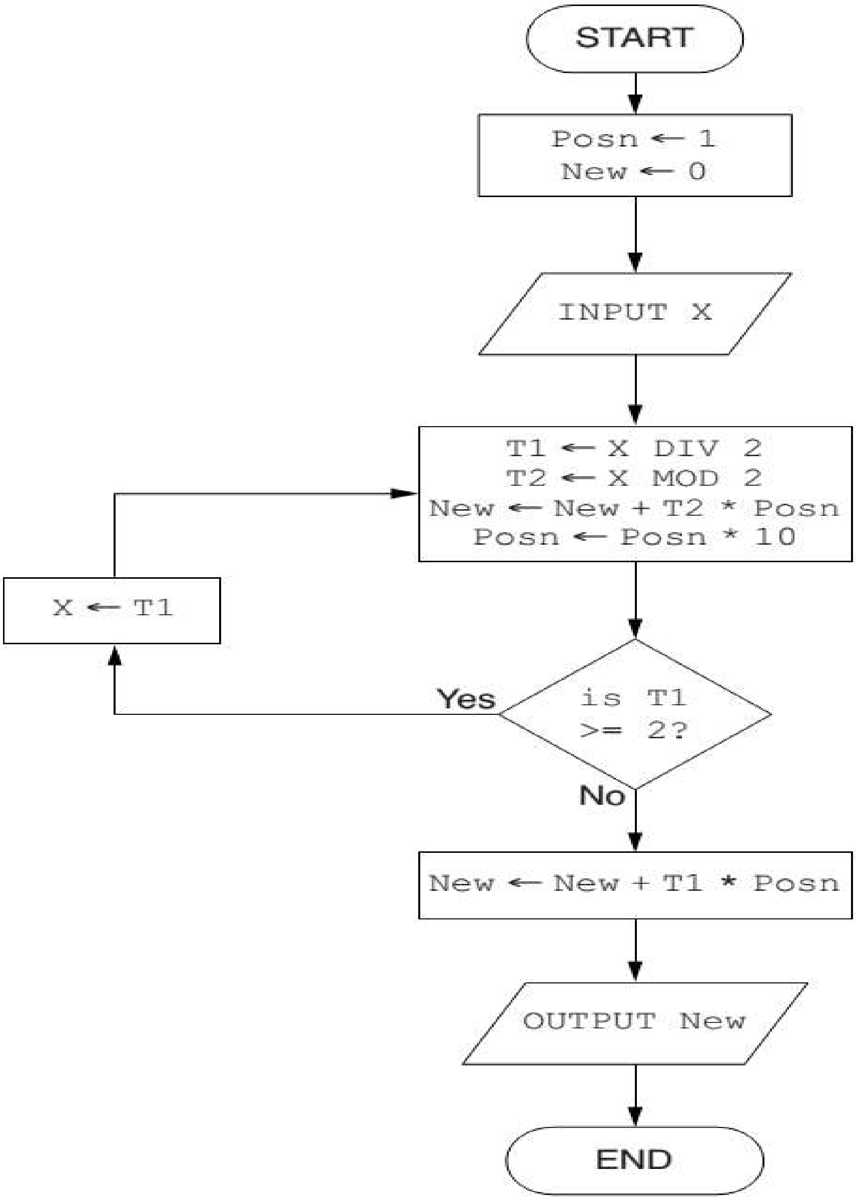
(b) Better candidates correctly identified the purpose of the flowchart as performing a check digit calculation.

(c) Very few candidates correctly identified a problem with the check digit calculation. The flowchart cannot deal with a remainder of 10 from the check digit calculation, there needs to be a special case where x is used as the check digit.

Q 11.30 Winter 2015 P21 & 22

3 (a) The flowchart inputs an integer. The predefined function DIV gives the integer result of the division, e.g. Y 10 DIV 3 gives the value Y = 3. The predefined function MOD gives the value of the remainder, e.g. Y 10 MOD 3 gives the value Y = 1.

Trace table for input value 5



X	Posn	New	T1	T2	OUTPUT

Trace table for input value 12

X	Posn	New	T1	T2	OUTPUT

(b) State the purpose of the flowchart in **part (a)**.

....[1]

Q 11.31 Winter 2015 P23

3 (a) This pseudo code inputs an integer. The predefined function DIV gives the value of the division, e.g. Y 10 DIV 3 gives the value Y = 3. The predefined function MOD gives the value of the remainder, e.g. Y 10 MOD 3 gives the value Y = 1.gives the value Y = 1.

```
INPUT X
WHILE X > 15
  DO
  T1 ← X DIV 16
  T2 ← X MOD 16
  CASE T2 OF
    10:OUTPUT A
    11:OUTPUT B
    12:OUTPUT C
    13:OUTPUT D
    14:OUTPUT E
    15:OUTPUT F
    OTHERWISE OUTPUT T2
  ENDCASE
  X ← T1
ENDWHILE
CASE X OF
  10:OUTPUT A
  11:OUTPUT B
  12:OUTPUT C
  13:OUTPUT D
  14:OUTPUT E
  15:OUTPUT F
  OTHERWISE OUTPUT X
ENDCASE
```

Complete a trace table for each of the **two** input values 37 and 191.

Trace table for input value 37

X	T1	T2	OUTPUT

Trace table for input value 191

X	T1	T2	OUTPUT

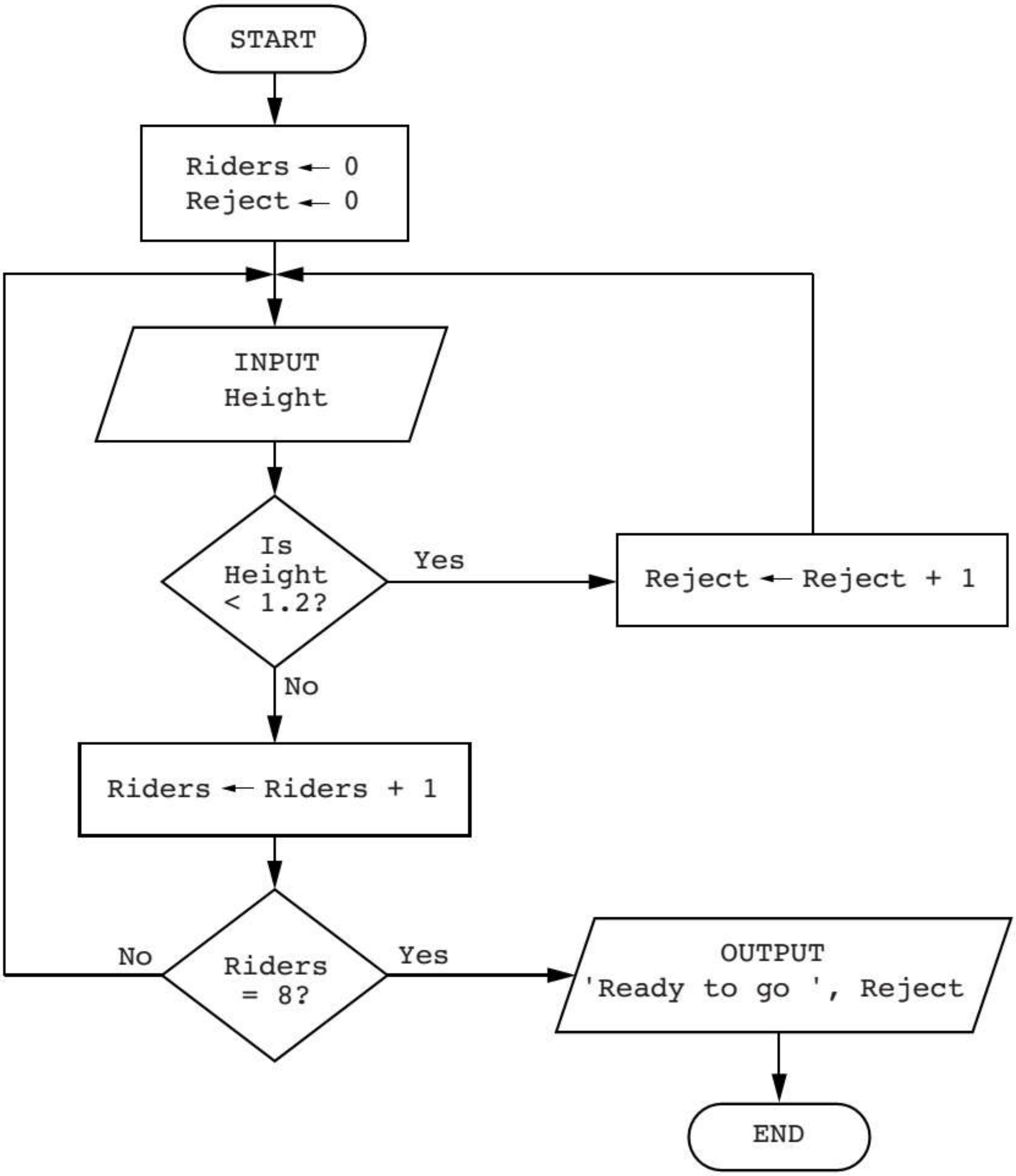
(b) State the purpose of the pseudo code in **part (a)**.

....

....[2]

Q 11.32 Summer 20162210,0478 P21 &P23

4 The flowchart below inputs the height of children who want to ride on a rollercoaster. Children under 1.2 metres are rejected. The ride starts when eight children have been accepted.



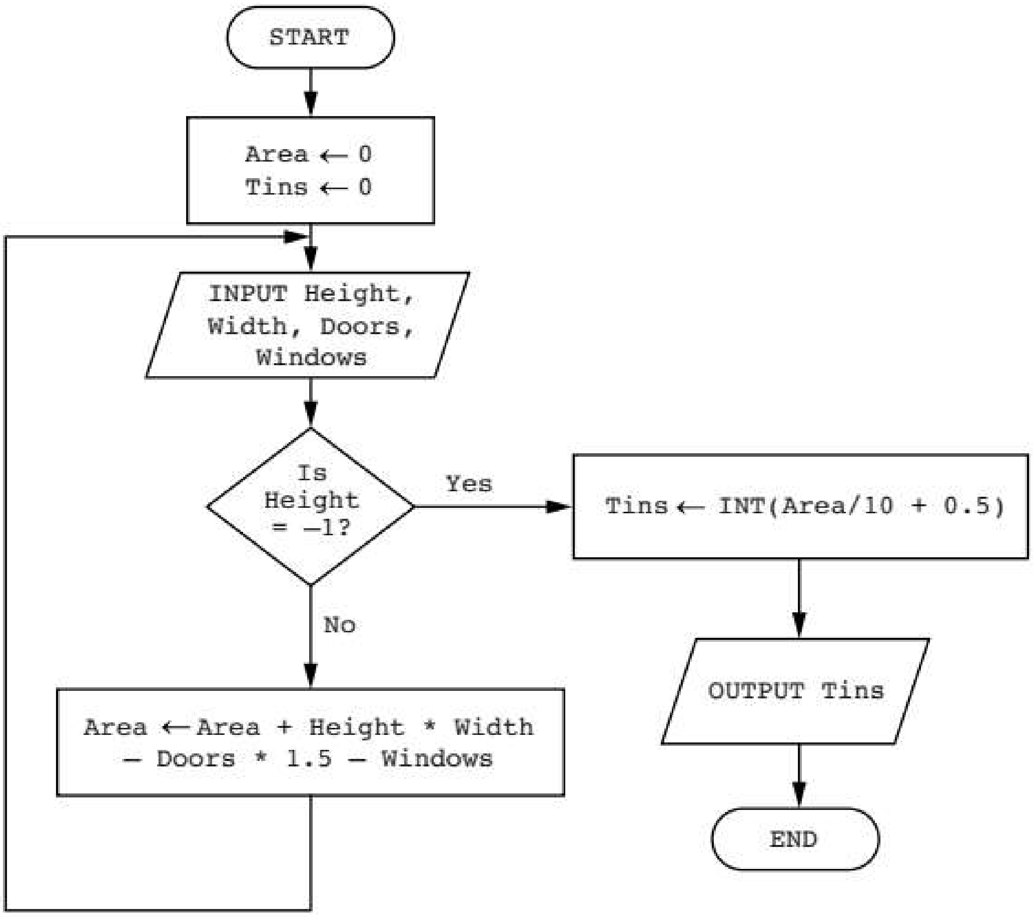
Complete the trace table for the input data:

1.4, 1.3, 1.1, 1.3, 1.0, 1.5, 1.2, 1.3, 1.4, 1.3, 0.9, 1.5, 1.6, 1.0 [4]

Riders	Reject	Height	OUTPUT

Q 11.33 Summer 20162210,0478 P22

3 The flowchart below calculates the number of tins of paint required to paint walls. The flowchart inputs the height and width of a wall in metres, the number of doors and the number of windows. A value of ‑1 for the height stops the input.


START
Area ← 0
Tins ← 0
INPUT Height, Width, Doors, Windows
Is Height = –1?
Yes
Tins ← INT(Area/10 + 0.5)
OUTPUT Tins
END
No
Area ← Area + Height * Width – Doors * 1.5 – Windows

Complete the trace table for the input data:

3, 5, 1, 0, 3, 7, 0, 0, 3, 5, 0, 3, 3, 7, 1, 1, –1, 0, 0, 0

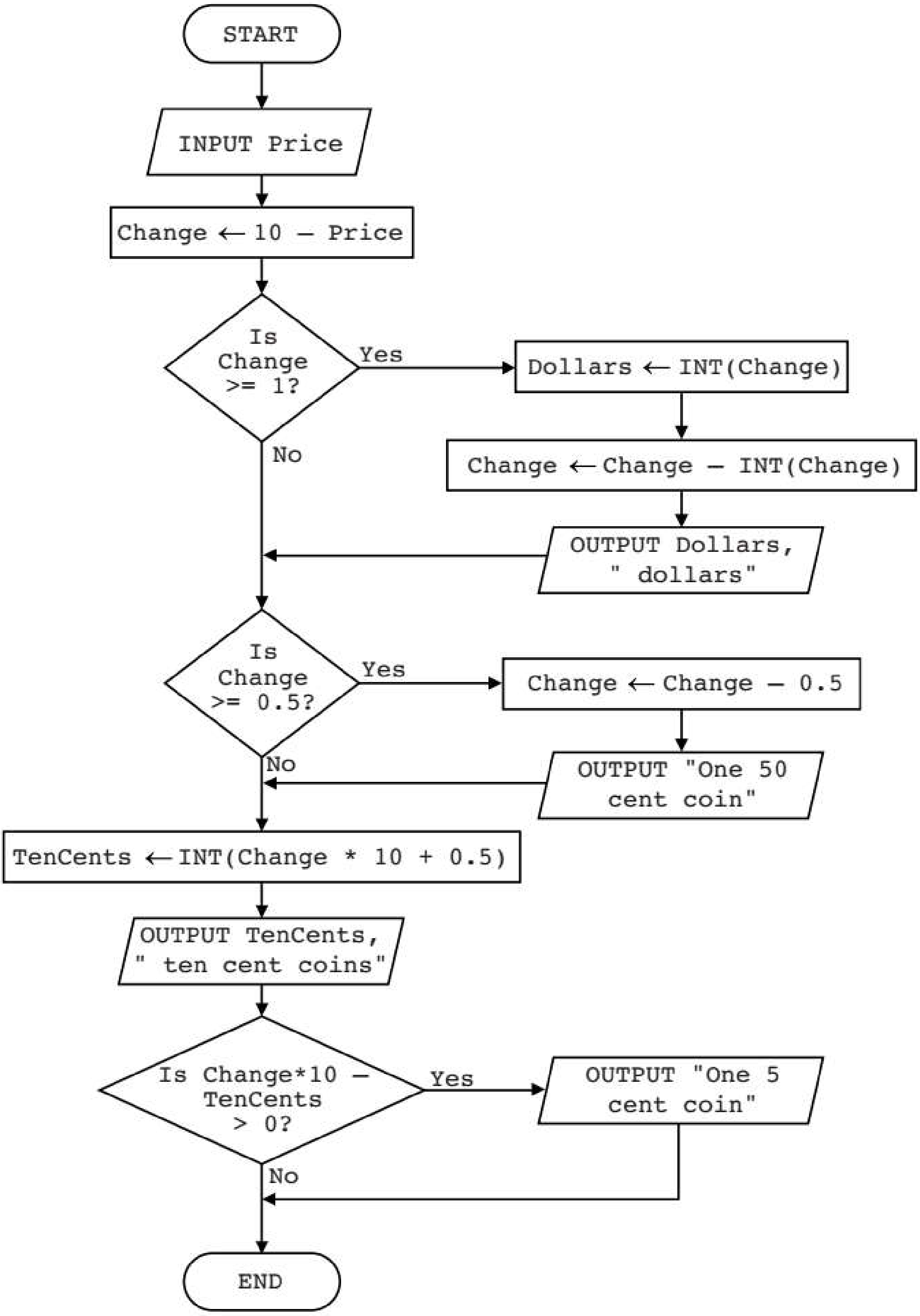
Area	Tins	Height	Width	Doors	Windows

[4]

Q 11.34 Winter 2016 P21-23

3 The flowchart below inputs the price of an item under $10. The change from a $10 note is output. Any amount less than 5 cents is rounded up to 5 cents.

The predefined function INT rounds a number down to the nearest whole number; for example Z ←INT(5.7) gives the value Z = 5

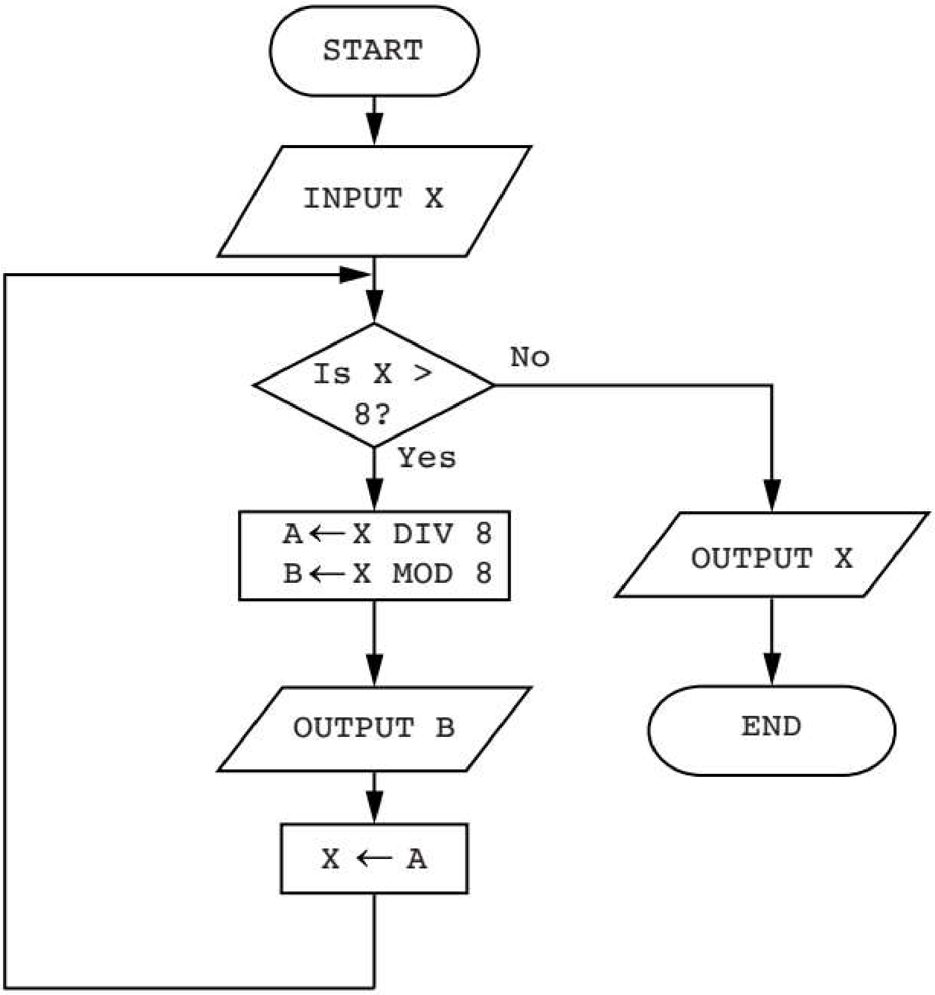

START
INPUT Price
Change ← 10 – Price
Is Change >= 1?
Yes
Dollars ← INT(Change)
Change ← Change – INT(Change)
OUTPUT Dollars, " dollars"
No
Is Change >= 0.5?
Yes
Change ← Change – 0.5
OUTPUT "One 50 cent coin"
No
TenCents ← INT(Change * 10 + 0.5)
OUTPUT TenCents, " ten cent coins"
Is Change*10 – TenCents > 0?
Yes
OUTPUT "One 5 cent coin"
No
END

Complete the trace table for the input data: 6.29 [5]

Price	Change	Dollars	TenCents	OUTPUT

Q 11.35 Winter 2016 P22

3 The flowchart below inputs an integer. The predefined function DIV gives the value of the division, for example Z ←11 DIV 3 gives the value Z = 3. The predefined function MOD gives the value of the remainder, for example Z ←11 MOD 3 gives the value Z = 2.



Complete a trace table for each of the two input values **33** and **75**. [4]

Trace table for input value **33**

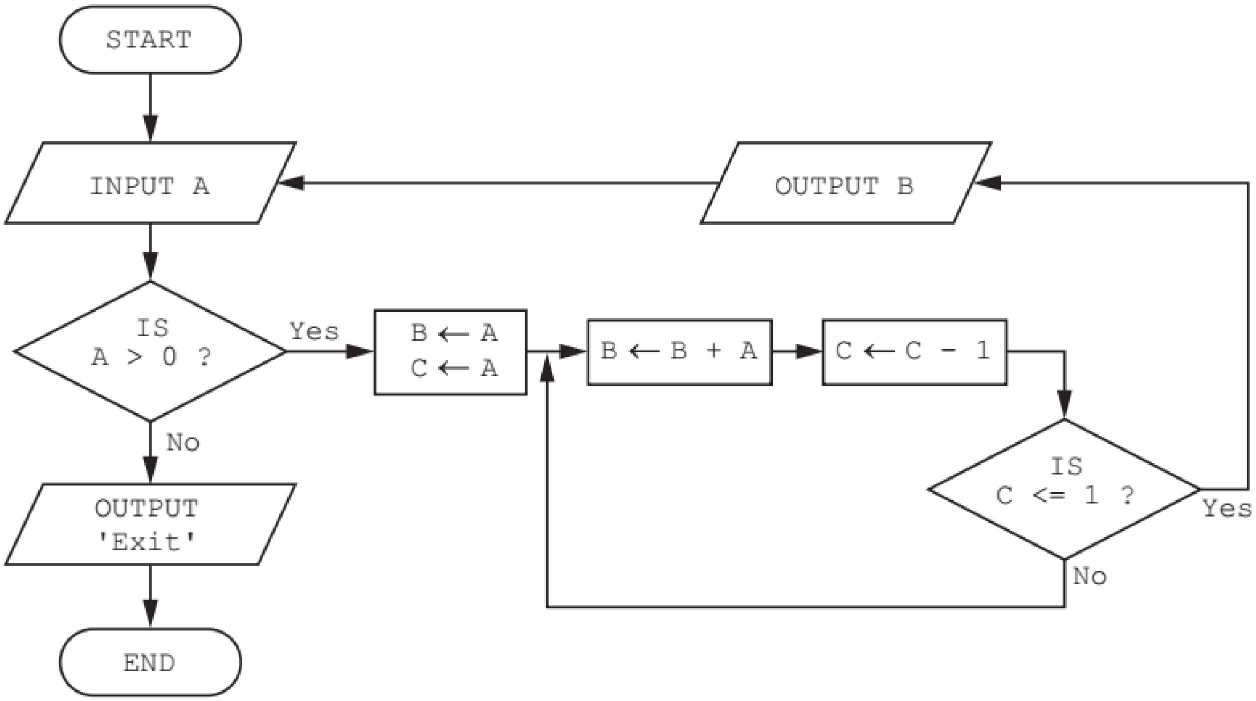
X	A	B	OUTPUT

Trace table for input value **75**

X	A	B	OUTPUT

Q 11/36 March 2017 P21 (India)

4 Study the flowchart.



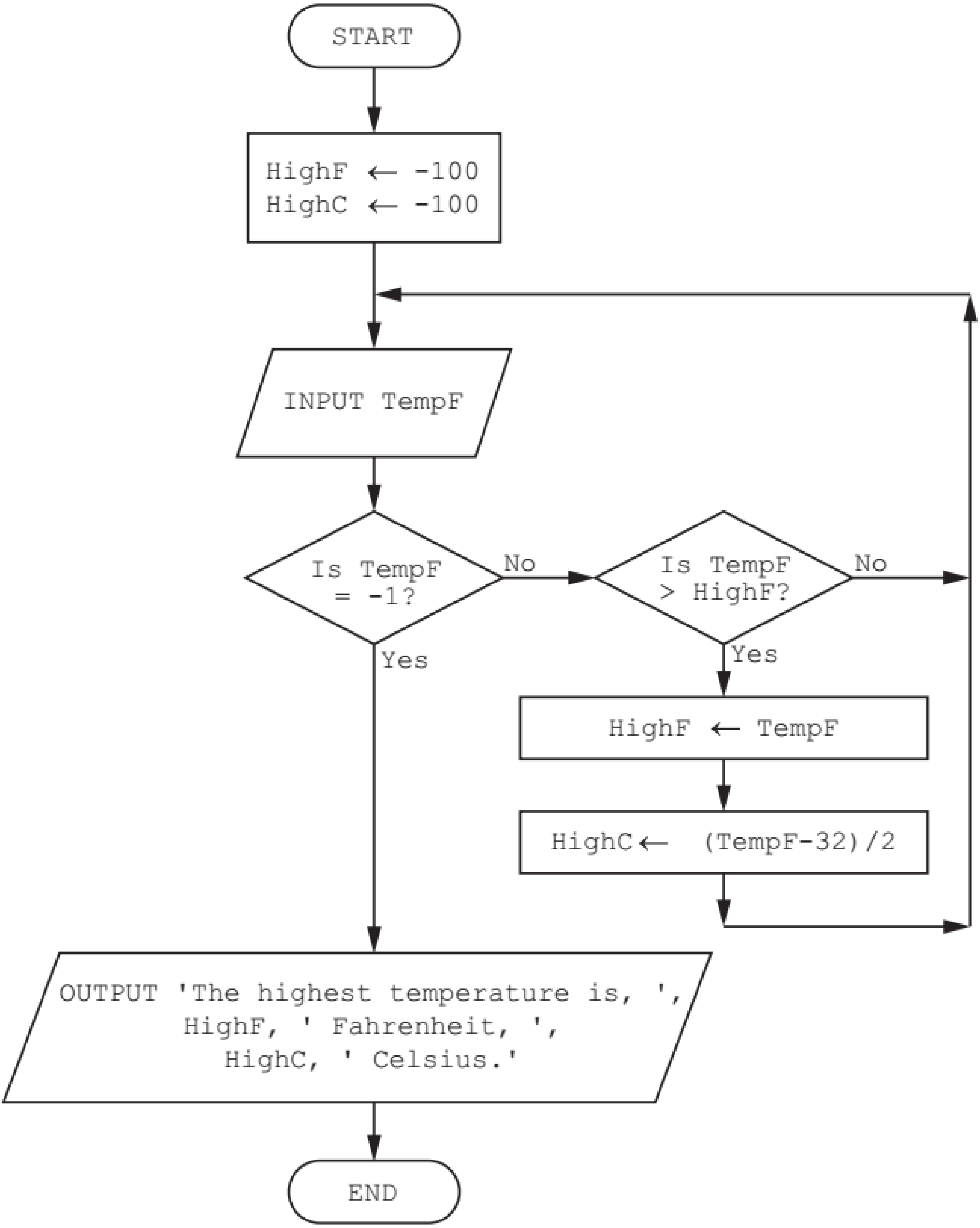
Complete the trace table for the input values 4, 3, -1: [4]

A	**B**	**C**	**OUTPUT**

Q 11.37 Summer 2017 P21

6 This flowchart inputs a range of temperatures in degrees Fahrenheit.

As each temperature is input, it is compared with the previous highest temperature. If it is higher than the current highest, it replaces the previous highest temperature and then it is converted to degrees Celsius.For ease of calculation, the final step of the Fahrenheit to Celsius conversion has been approximated as division by 2.When –1 is entered, the input process stops and the highest temperature (in both Fahrenheit and Celsius) is output.

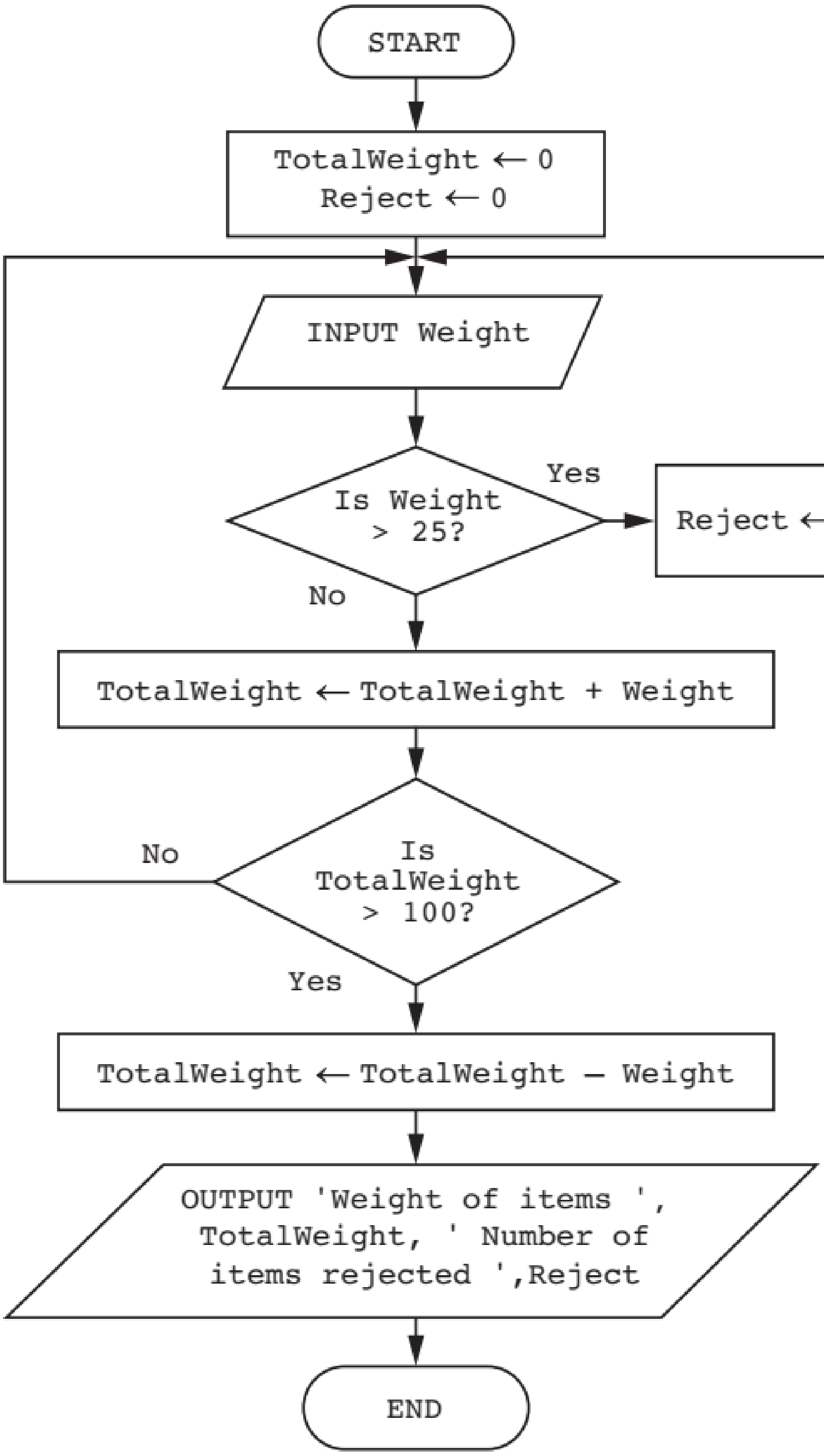


Complete the trace table for the input data:
68, 46, 50, 86, 65, 50, 40, 30, –1 [5]

HighF	HighC	TempF	OUTPUT

Q 11.38 Summer 2017 P22

3 This flowchart inputs the weight of items in kilograms to be loaded on a trailer. Any item over25 kilograms is rejected. The trailer can take up to 100 kilograms.

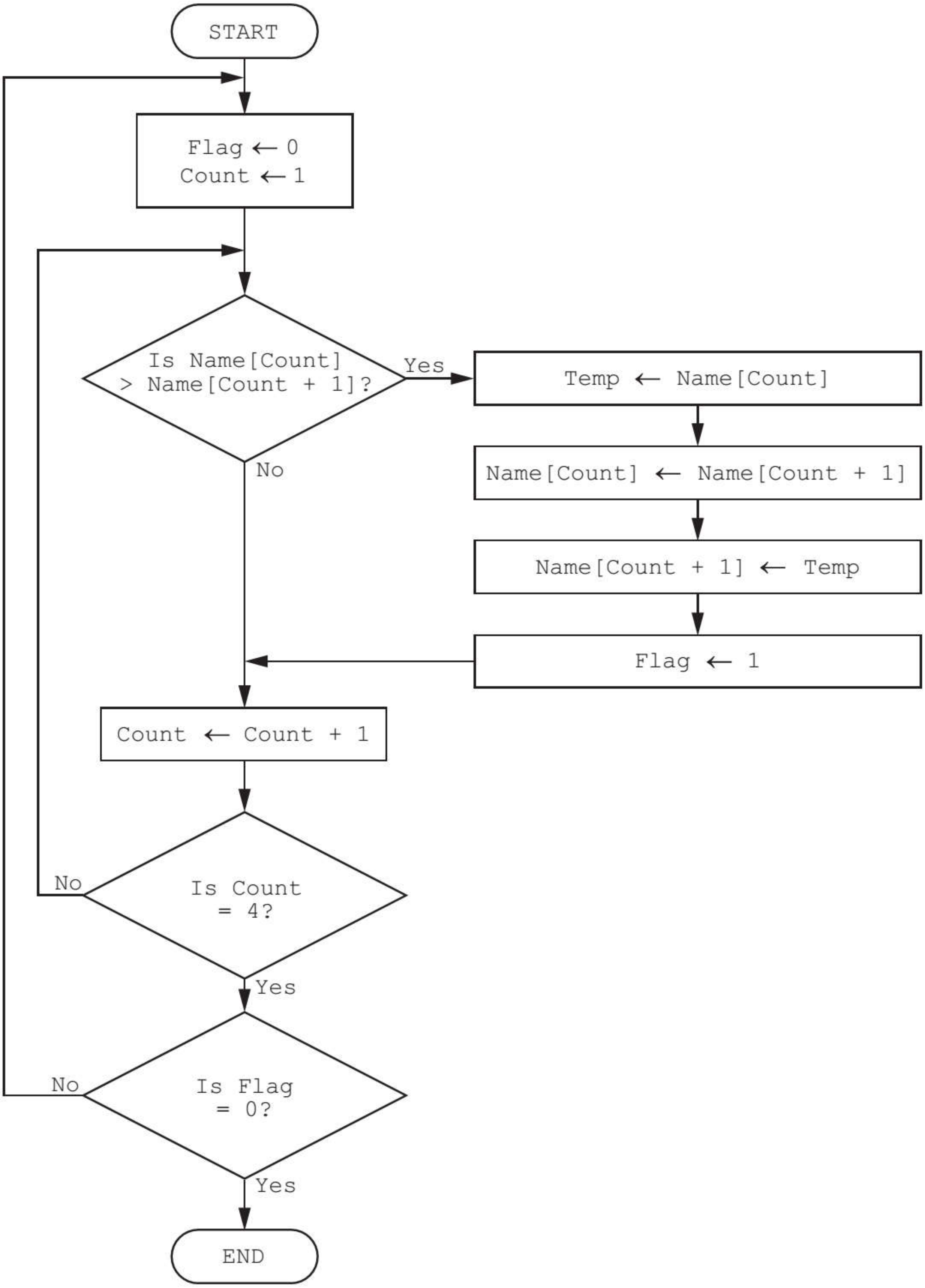

START
TotalWeight ← 0
Reject ← 0
INPUT Weight
Is Weight > 25?
Yes
Reject ← Reject + 1
No
TotalWeight ← TotalWeight + Weight
No
Is TotalWeight > 100?
Yes
TotalWeight ← TotalWeight – Weight
OUTPUT 'Weight of items ', TotalWeight, ' Number of items rejected ',Reject
END

Complete the trace table for the input data:
13, 17, 26, 25, 5, 10, 15, 35, 20, 15 [5]

Weight	Reject	TotalWeight	OUTPUT

Q 11.39 Winter 2017 P21

5 The flowchart below represents a program routine.



(a) The array used in the flowchart contains the following data:

Name[1]	**Name[2]**	**Name[3]**	**Name[4]**
Jamal	Amir	Eve	Tara

Complete the trace table using the data given in the array. [5]

Flag	**Count**	**Name[1]**	**Name[2]**	**Name[3]**	**Name[4]**	**Temp**
		Jamal	Amir	Eve	Tara	

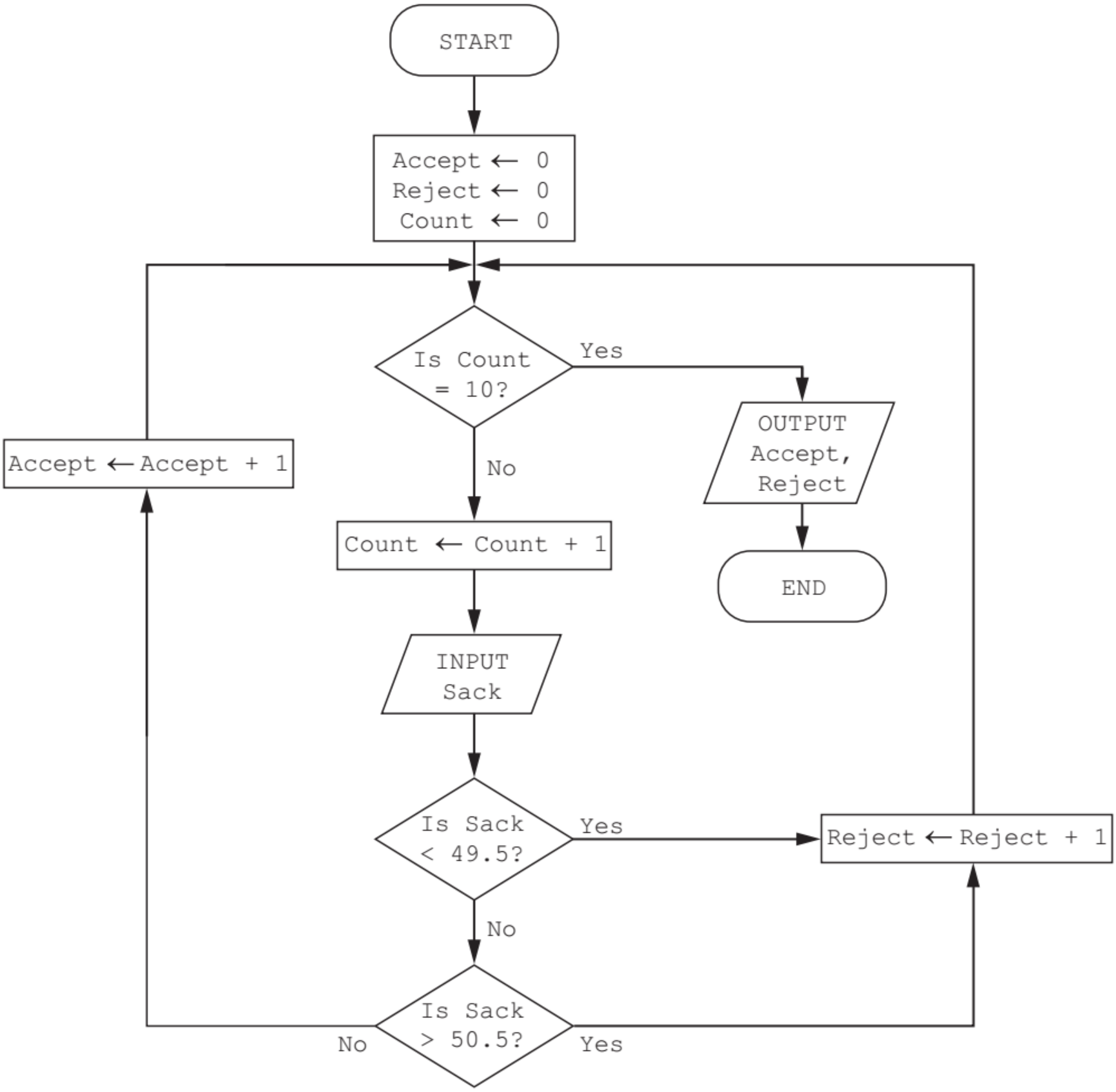
(b) Describe what the algorithm represented by the flowchart is doing.

....

....[2]

Q 11.40 Winter 2017 P22

5 (a) This flowchart checks a batch of 10 rice sacks for weight. Sacks should weigh 50 kilograms each. Sacks weighing over 50.5 kilograms or less than 49.5 kilograms are rejected. The number of sacks accepted and the number of sacks rejected is output.



Complete the trace table for the input data:

50.4, 50.3, 49.1, 50.3, 50.0, 49.5, 50.2, 50.3, 50.5, 50.6 [5]

Accept	Reject	Count	Sack	OUTPUT

(b) The size of the batch has increased to 50 sacks. It has been decided to only reject sacks thatare underweight.

State the changes that need to be made to the flowchart.

....

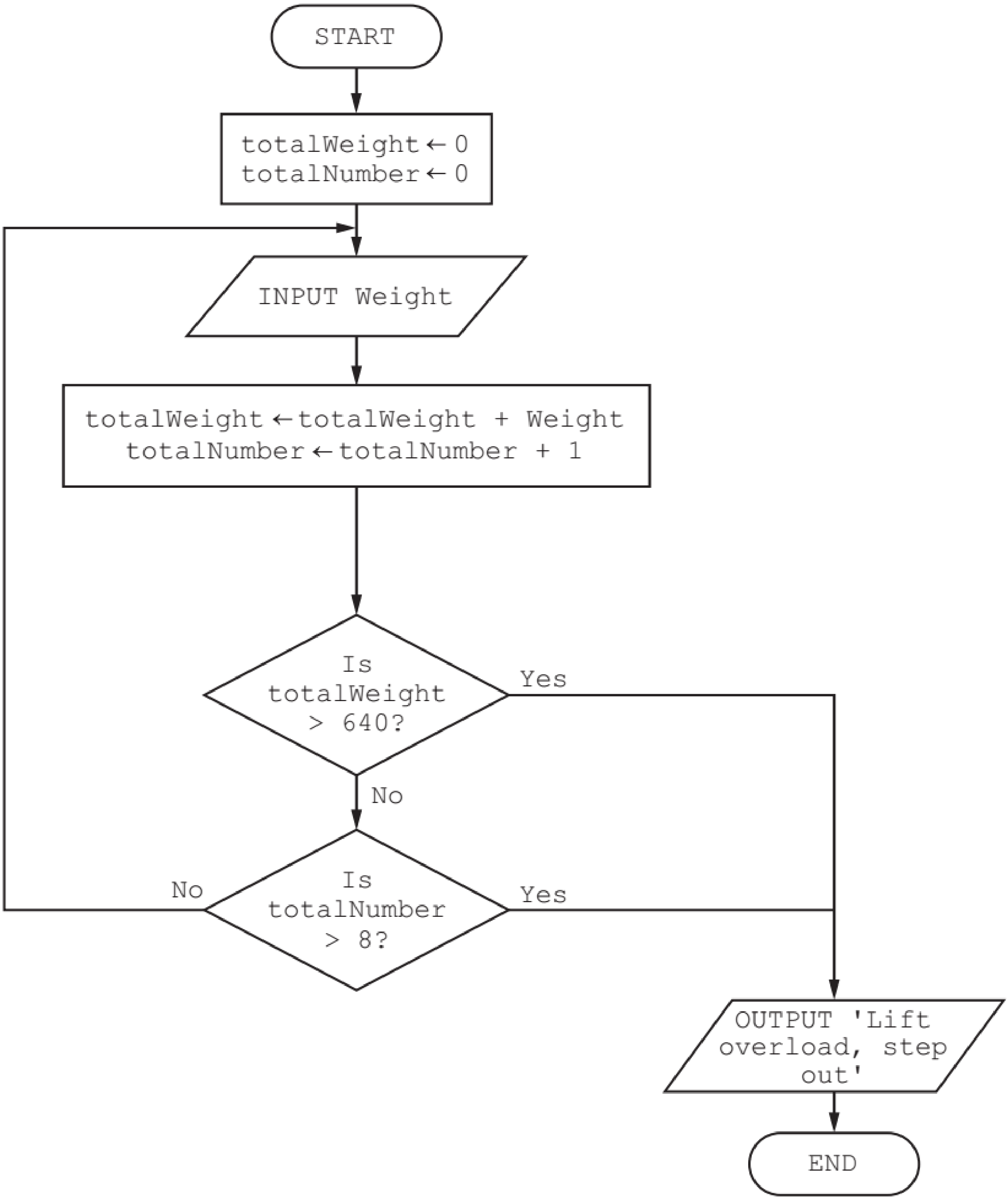
....

....

....[2]

Q 11.41 March 2018 P22 (India)

3 This flowchart inputs the weight in kilograms of a passenger stepping into a lift. The lift can take a maximum of eight passengers or a maximum weight of 640 kilograms.



Complete the trace table for the passenger input data: [4]
50, 70, 65, 100, 95, 50, 55, 85, 70, 75

Weight	TotalWeight	TotalNumber	OUTPUT

Comments on Question 3

Most candidates showed the skill of using a trace table. Some candidates provided a 'rough answer' in penciland a final answer in ink; this is not recommended as extra values can be seen in the trace table. Candidatesfound the output the most challenging column to complete correctly and common errors seen were toincorrectly include quotation marks around the output or repeats of the output.

Q 11.42 Summer 2018 P21

3 The global trade item number (GTIN-8) barcode has seven digits and a check digit.
This pseudocode algorithm inputs seven digits and calculates the eighth digit, then outputs theGTIN-8.
DIV(X,Y), finds the number of divides in division for example **DIV(23,10)** is 2.
MOD(X,Y), finds the remainder in division for example **MOD(23,10)** is 3.

```
FOR Count ← 1 TO 7
   INPUT Number
   Digit(Count) ← Number
NEXT
Sum ← (Digit(1)+Digit(3)+Digit(5)+Digit(7))*3+Digit(2)+Digit(4)+Digit(6)
IF MOD(Sum,10) <> 0
   THEN Digit(8) ← DIV(Sum,10)*10 + 10 - Sum
   ELSE Digit(8) ← 0
ENDIF
OUTPUT "GTIN-8"
FOR Count ← 1 TO 8
   OUTPUT Digit(Count)
NEXT
```

(a) Complete the trace table for the input data: 5, 7, 0, 1, 2, 3, 4 [5]

Digit (1)	Digit (2)	Digit (3)	Digit (4)	Digit (5)	Digit (6)	Digit (7)	Digit (8)	Sum	OUTPUT

Complete the trace table for the input data: 4, 3, 1, 0, 2, 3, 1

Digit (1)	Digit (2)	Digit (3)	Digit (4)	Digit (5)	Digit (6)	Digit (7)	Digit (8)	Sum	OUTPUT

(b) Explain how you would change the algorithm to input eight digits (seven digits and the checkdigit) and output if the check digit entered is correct or not.

....

....

....

....

....

....

.... [3]

Q 11.43 Summer 2018 P22

3 This pseudocode algorithm inputs two non-zero numbers and a sign, and then performs thecalculation shown by the sign. An input of zero for the first number terminates the process.

```
INPUT Number1, Number2, Sign
WHILE Number1 <> 0
   IF Sign = '+' THEN Answer ← Number1 + Number2 ENDIF
   IF Sign = '-' THEN Answer ← Number1 - Number2 ENDIF
   IF Sign = '*' THEN Answer ← Number1 * Number2 ENDIF
   IF Sign = '/' THEN Answer ← Number1 / Number2 ENDIF
   IF Sign <> '/' AND Sign <> '*' AND Sign <> '-' AND Sign <> '+'
      THEN Answer ← 0
   ENDIF
   IF Answer <> 0 THEN OUTPUT Answer ENDIF
   INPUT Number1, Number2, Sign
ENDWHILE
```

(a) Complete the trace table for the input data:

5, 7, +, 6, 2, –, 4, 3, *, 7, 8, ?, 0, 0, / [3]

Number1	Number2	Sign	Answer	OUTPUT

(b) Show how you could improve the algorithm written in pseudocode by writing analternativetype of conditional statement in pseudocode.

....

....

....

....

....

....

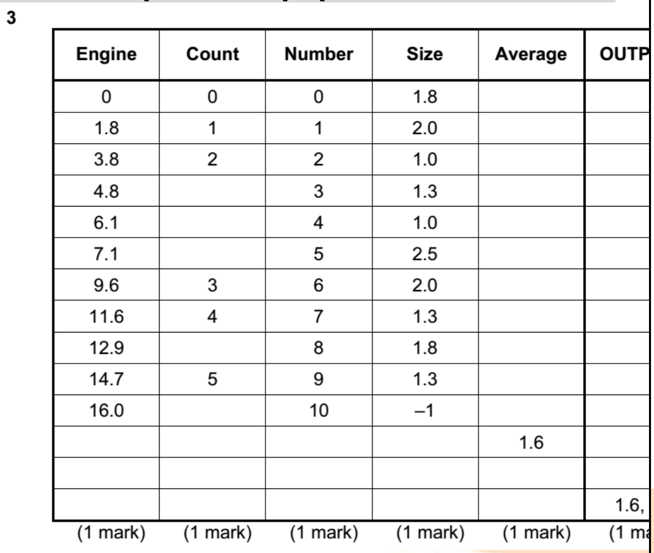
....

....

....

....[3]

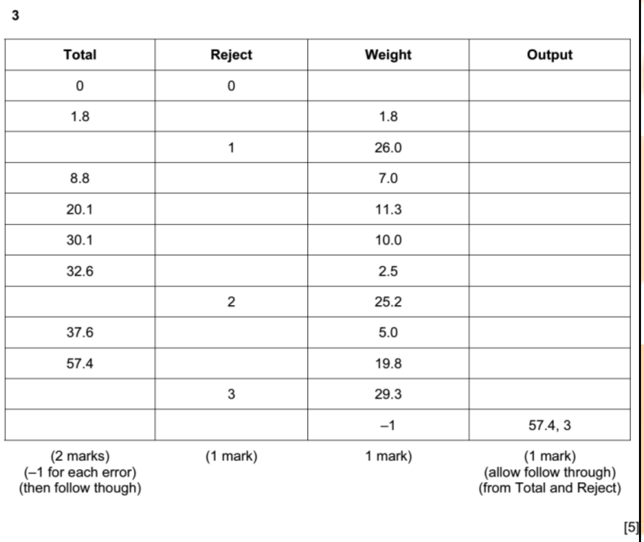
Q 11.27 Specimen paper P2



3

Engine	Count	Number	Size	Average	OUTP
0	0	0	1.8		
1.8	1	1	2.0		
3.8	2	2	1.0		
4.8		3	1.3		
6.1		4	1.0		
7.1		5	2.5		
9.6	3	6	2.0		
11.6	4	7	1.3		
12.9		8	1.8		
14.7	5	9	1.3		
16.0		10	–1		
				1.6	
					1.6,
(1 mark)	(1 mark)	(1 mark)	(1 mark)	(1 mark)	(1 ma

Q 11.28 Summer 2015 P21& 23

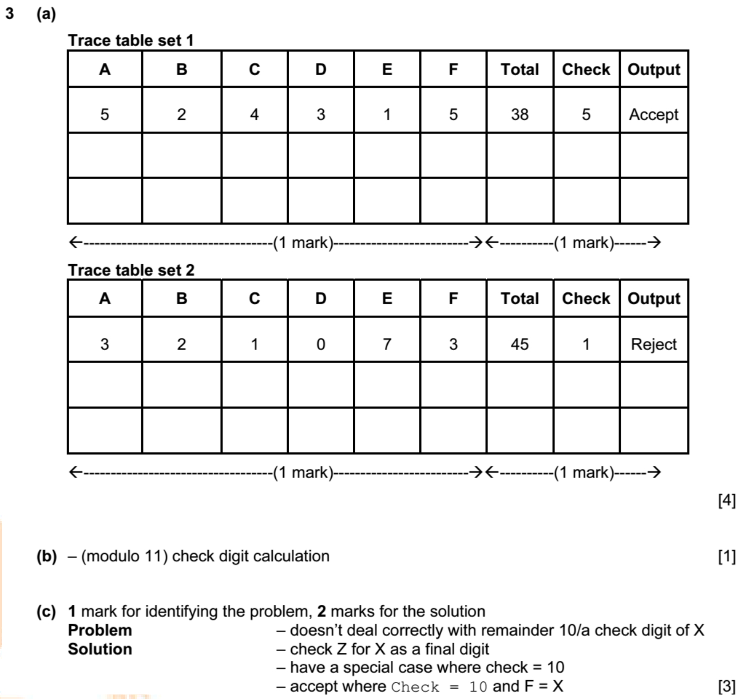


3

Total	Reject	Weight	Output
0	0		
1.8		1.8	
	1	26.0	
8.8		7.0	
20.1		11.3	
30.1		10.0	
32.6		2.5	
	2	25.2	
37.6		5.0	
57.4		19.8	
	3	29.3	
		–1	57.4, 3
(2 marks) (–1 for each error) (then follow though)	(1 mark)	1 mark)	(1 mark) (allow follow through) (from Total and Reject)

[5]

Q 11.29 Summer 2015 P22



3 (a)

Trace table set 1

A	B	C	D	E	F	Total	Check	Output
5	2	4	3	1	5	38	5	Accept

←-----------------------------------(1 mark)-------------------------→←----------(1 mark)------→

Trace table set 2

A	B	C	D	E	F	Total	Check	Output
3	2	1	0	7	3	45	1	Reject

←-----------------------------------(1 mark)-------------------------→←----------(1 mark)------→

[4]

(b) – (modulo 11) check digit calculation [1]

(c) **1** mark for identifying the problem, **2** marks for the solution

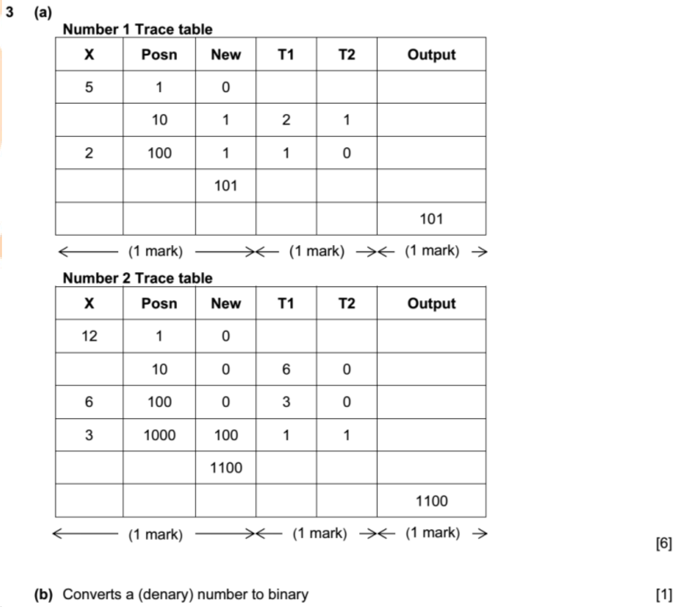
Problem – doesn't deal correctly with remainder 10/a check digit of X

Solution – check Z for X as a final digit

– have a special case where check = 10

– accept where `Check = 10` and F = X [3]

Q 11.30 Winter 2015 P21 & 22



3 (a)

Number 1 Trace table

X	Posn	New	T1	T2	Output
5	1	0			
	10	1	2	1	
2	100	1	1	0	
		101			
					101

← (1 mark) →← (1 mark) →← (1 mark) →

Number 2 Trace table

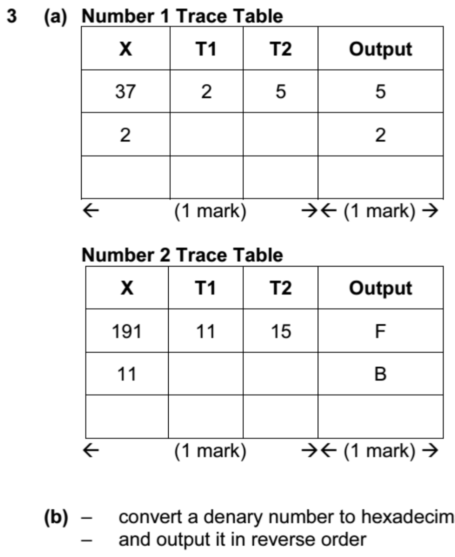
X	Posn	New	T1	T2	Output
12	1	0			
	10	0	6	0	
6	100	0	3	0	
3	1000	100	1	1	
		1100			
					1100

← (1 mark) →← (1 mark) →← (1 mark) →

[6]

(b) Converts a (denary) number to binary [1]

Q 11.31 Winter 2015 P23



3 (a) Number 1 Trace Table

X	T1	T2	Output
37	2	5	5
2			2

← (1 mark) →← (1 mark) →

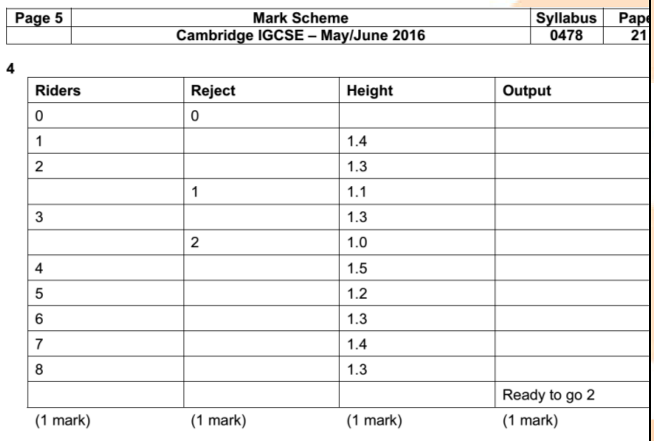
Number 2 Trace Table

X	T1	T2	Output
191	11	15	F
11			B

← (1 mark) →← (1 mark) →

(b) – convert a denary number to hexadecimal
– and output it in reverse order

Q 11.32 Summer 2016 P21 &P23

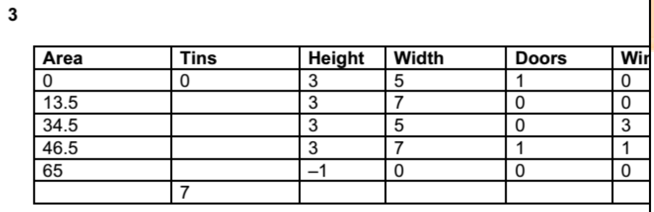


Page 5	Mark Scheme Cambridge IGCSE – May/June 2016	Syllabus 0478	Pape 21

4

Riders	Reject	Height	Output
0	0		
1		1.4	
2		1.3	
	1	1.1	
3		1.3	
	2	1.0	
4		1.5	
5		1.2	
6		1.3	
7		1.4	
8		1.3	
			Ready to go 2
(1 mark)	(1 mark)	(1 mark)	(1 mark)

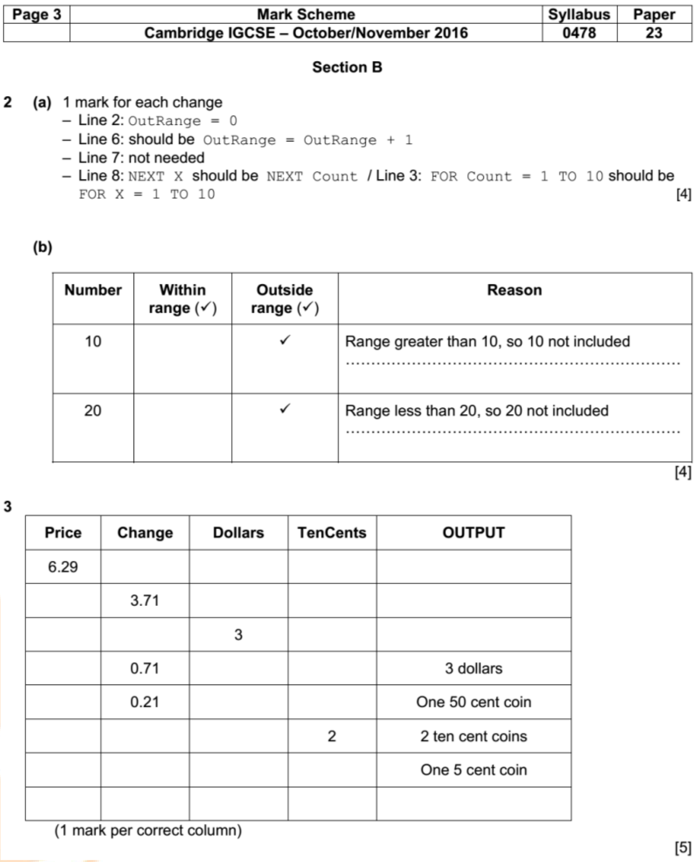
Q 11.33 Summer 20162210,0478 P22



3

Area	Tins	Height	Width	Doors	Wir
0	0	3	5	1	0
13.5		3	7	0	0
34.5		3	5	0	3
46.5		3	7	1	1
65		–1	0	0	0
	7				

Q 11.34 Winter 2016 P21-23



Page 3	Mark Scheme Cambridge IGCSE – October/November 2016	Syllabus 0478	Paper 23

Section B

2 (a) 1 mark for each change
– Line 2: `OutRange = 0`
– Line 6: should be `OutRange = OutRange + 1`
– Line 7: not needed
– Line 8: `NEXT X` should be `NEXT Count` / Line 3: `FOR Count = 1 TO 10` should be `FOR X = 1 TO 10` [4]

(b)

Number	Within range (✓)	Outside range (✓)	Reason
10		✓	Range greater than 10, so 10 not included
20		✓	Range less than 20, so 20 not included

[4]

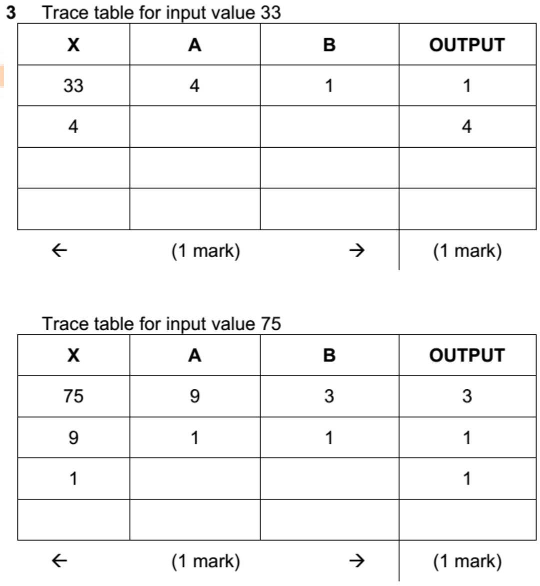
3

Price	Change	Dollars	TenCents	OUTPUT
6.29				
	3.71			
		3		
	0.71			3 dollars
	0.21			One 50 cent coin
			2	2 ten cent coins
				One 5 cent coin

(1 mark per correct column)

[5]

Q 11.35 Winter 2016 P22



3 Trace table for input value 33

X	A	B	OUTPUT
33	4	1	1
4			4

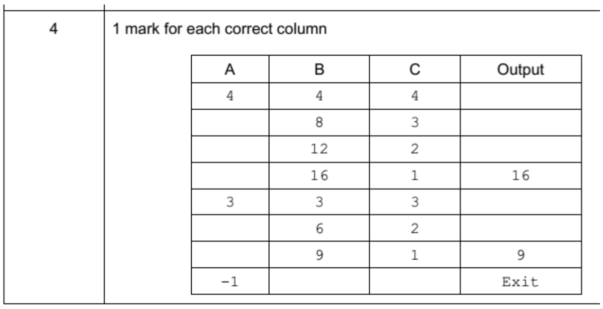
← (1 mark) → (1 mark)

Trace table for input value 75

X	A	B	OUTPUT
75	9	3	3
9	1	1	1
1			1

← (1 mark) → (1 mark)

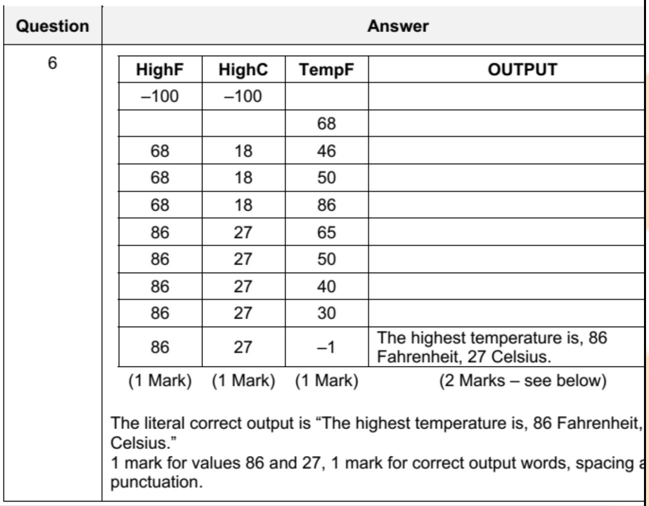
Q 11.36 March 2017 P21 (India)



4	1 mark for each correct column

A	B	C	Output
4	4	4	
	8	3	
	12	2	
	16	1	16
3	3	3	
	6	2	
	9	1	9
-1			Exit

Q 11.37 Summer 2017 P21

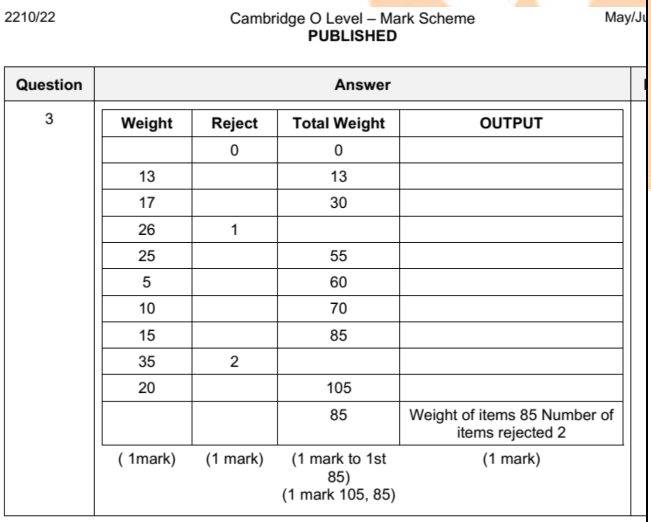


Question	Answer
6	

HighF	HighC	TempF	OUTPUT
–100	–100		
		68	
68	18	46	
68	18	50	
68	18	86	
86	27	65	
86	27	50	
86	27	40	
86	27	30	
86	27	–1	The highest temperature is, 86 Fahrenheit, 27 Celsius.
(1 Mark)	(1 Mark)	(1 Mark)	(2 Marks – see below)

The literal correct output is "The highest temperature is, 86 Fahrenheit, Celsius."
1 mark for values 86 and 27, 1 mark for correct output words, spacing a punctuation.

Q 11.38 Summer 2017 P22

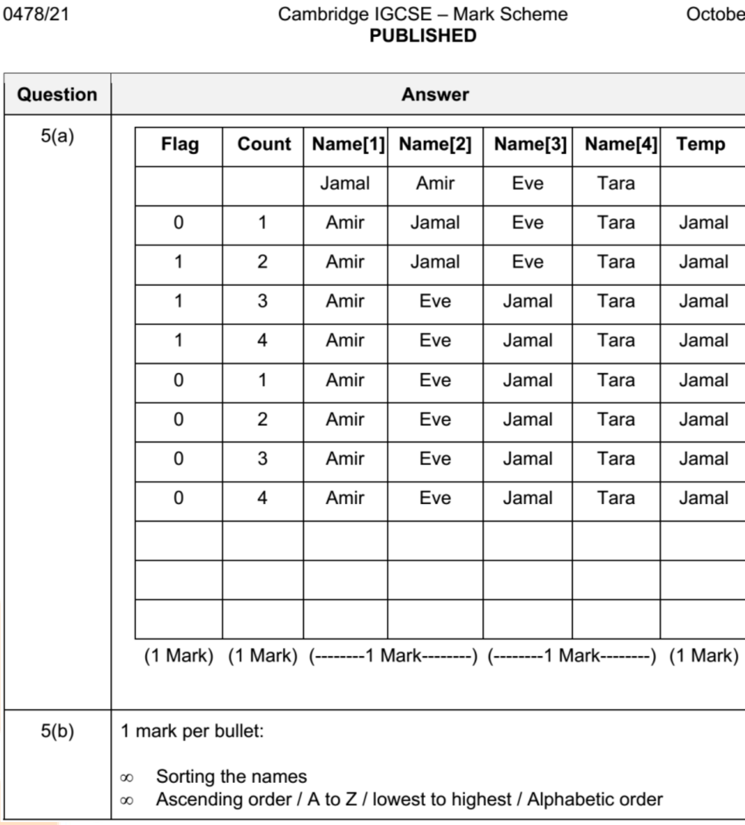


2210/22 Cambridge O Level – Mark Scheme May/J
PUBLISHED

Question	Answer
3	

Weight	Reject	Total Weight	OUTPUT
	0	0	
13		13	
17		30	
26	1		
25		55	
5		60	
10		70	
15		85	
35	2		
20		105	
		85	Weight of items 85 Number of items rejected 2
(1mark)	(1 mark)	(1 mark to 1st 85) (1 mark 105, 85)	(1 mark)

Q 11.39 Winter 2017 P21



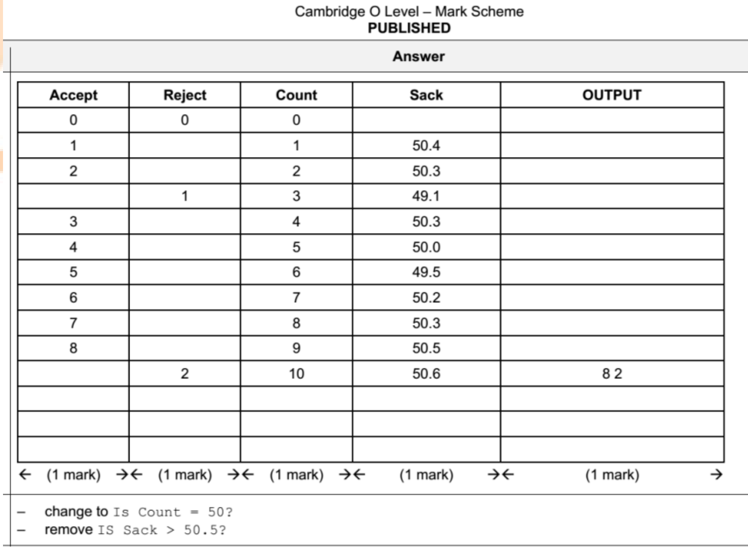
0478/21 Cambridge IGCSE – Mark Scheme Octobe
PUBLISHED

Question	Answer
5(a)	

Flag	Count	Name[1]	Name[2]	Name[3]	Name[4]	Temp
		Jamal	Amir	Eve	Tara	
0	1	Amir	Jamal	Eve	Tara	Jamal
1	2	Amir	Jamal	Eve	Tara	Jamal
1	3	Amir	Eve	Jamal	Tara	Jamal
1	4	Amir	Eve	Jamal	Tara	Jamal
0	1	Amir	Eve	Jamal	Tara	Jamal
0	2	Amir	Eve	Jamal	Tara	Jamal
0	3	Amir	Eve	Jamal	Tara	Jamal
0	4	Amir	Eve	Jamal	Tara	Jamal
(1 Mark)	(1 Mark)	(--------1 Mark--------)		(--------1 Mark--------)		(1 Mark)

Question	Answer
5(b)	1 mark per bullet: ∞ Sorting the names ∞ Ascending order / A to Z / lowest to highest / Alphabetic order

Q 11.40 Winter 2017 P22



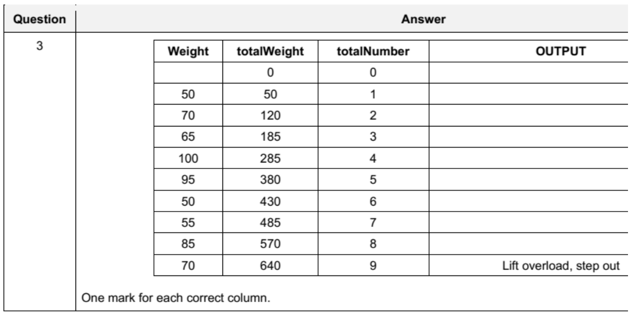
Cambridge O Level – Mark Scheme
PUBLISHED

Answer

Accept	Reject	Count	Sack	OUTPUT
0	0	0		
1		1	50.4	
2		2	50.3	
	1	3	49.1	
3		4	50.3	
4		5	50.0	
5		6	49.5	
6		7	50.2	
7		8	50.3	
8		9	50.5	
	2	10	50.6	8 2
← (1 mark) →	← (1 mark) →	← (1 mark) →	← (1 mark) →	← (1 mark) →

– change to `Is Count = 50?`
– remove `IS Sack > 50.5?`

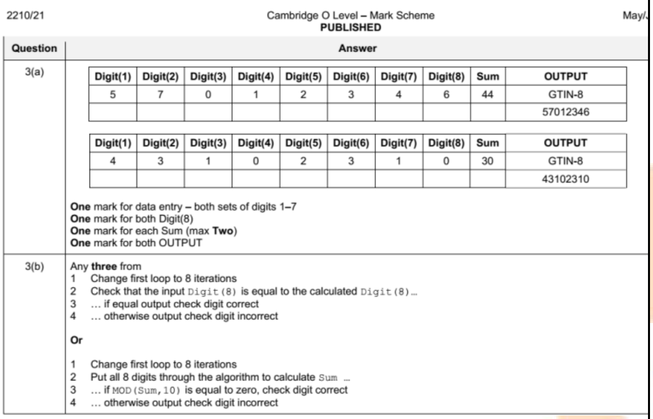
Q 11.41 March 2018 P22 (India)



Question	Answer			
3	**Weight**	**totalWeight**	**totalNumber**	**OUTPUT**
		0	0	
	50	50	1	
	70	120	2	
	65	185	3	
	100	285	4	
	95	380	5	
	50	430	6	
	55	485	7	
	85	570	8	
	70	640	9	Lift overload, step out

One mark for each correct column.

Q 11.42 Summer 2018 P21



2210/21 Cambridge O Level – Mark Scheme **PUBLISHED** May/

Question	Answer
3(a)	

Digit(1)	Digit(2)	Digit(3)	Digit(4)	Digit(5)	Digit(6)	Digit(7)	Digit(8)	Sum	OUTPUT
5	7	0	1	2	3	4	6	44	GTIN-8
									57012346

Digit(1)	Digit(2)	Digit(3)	Digit(4)	Digit(5)	Digit(6)	Digit(7)	Digit(8)	Sum	OUTPUT
4	3	1	0	2	3	1	0	30	GTIN-8
									43102310

One mark for data entry – both sets of digits 1–7
One mark for both Digit(8)
One mark for each Sum (max **Two**)
One mark for both OUTPUT

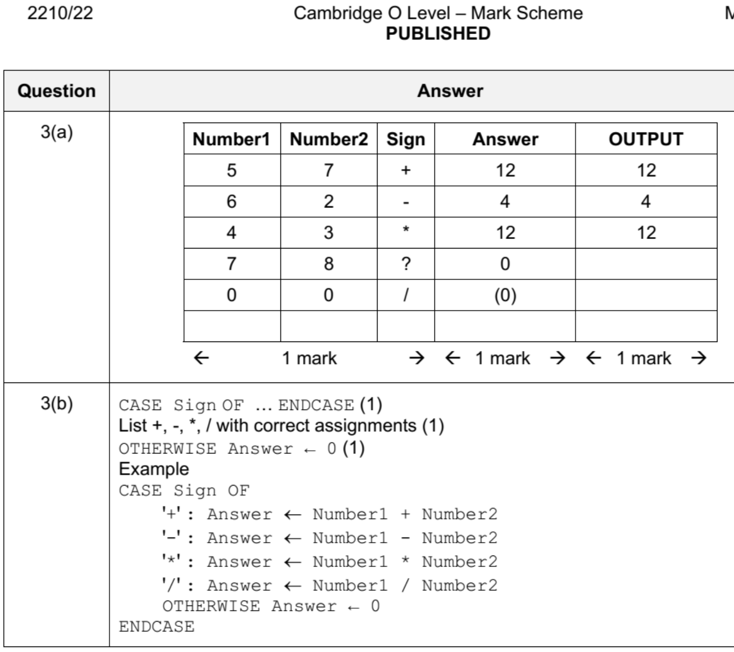
3(b)

Any **three** from
1 Change first loop to 8 iterations
2 Check that the input `Digit(8)` is equal to the calculated `Digit(8)`…
3 … if equal output check digit correct
4 … otherwise output check digit incorrect

Or

1 Change first loop to 8 iterations
2 Put all 8 digits through the algorithm to calculate `Sum` …
3 … if `MOD(Sum,10)` is equal to zero, check digit correct
4 … otherwise output check digit incorrect

Q 11.43 Summer 2018 P22



2210/22 Cambridge O Level – Mark Scheme **PUBLISHED** M

Question	Answer				
3(a)	**Number1**	**Number2**	**Sign**	**Answer**	**OUTPUT**
	5	7	+	12	12
	6	2	-	4	4
	4	3	*	12	12
	7	8	?	0	
	0	0	/	(0)	
	← 1 mark	→	← 1 mark →	← 1 mark →	

3(b)

`CASE Sign OF … ENDCASE` (1)
List +, -, *, / with correct assignments (1)
`OTHERWISE Answer ← 0` (1)
Example

```
CASE Sign OF
    '+' : Answer ← Number1 + Number2
    '-' : Answer ← Number1 - Number2
    '*' : Answer ← Number1 * Number2
    '/' : Answer ← Number1 / Number2
    OTHERWISE Answer ← 0
ENDCASE
```

Chapter 12

Algorithm pseudo code

Q 12.1) Summer 2006 (Extract)

A formula for calculating the body mass index (BMI) is:

$$\text{BMI} = \frac{\text{weight in kilograms}}{(\text{height in metres}) \times (\text{height in metres})}$$

Using pseudo code or otherwise, write an algorithm that will input weight (kg) and height (m) of students, calculate their body mass index (BMI) and output their BMI.

Test data: 80, 2, 100, 1.9, 60, 2, 70, 1.8

First draw trace table write down column headings

Calculate BMI using trace table:

Weight	Height	BMI

Setup in pseudo code using declaration of variable

SECTION SHOWS YOU HOW

Now Input Weight and height

Weight	Height	BMI
80	2	
100	1.9	
60	2	
70	1.8	

Input in pseudo code using test data

Now calculate the BMI using given formula

Weight	Height	BMI
80	2	20
100	1.9	28
60	2	15
70	1.8	22

Process in pseudo code using given formula

Now write down the above steps in pseudo code:

```
DECLARE Weight, Height, BMI: Real
INPUT Weight, Height
BMI ← Weight/(Height*Height)
OUTPUT BMI
```

Q12.2)Winter 2007 (Extract)

Fuel economy for a car is found using the formula:

$$\text{Fuel Economy} = \frac{\text{Distance Travelled (km)}}{\text{Fuel Used (litres)}}$$

Using pseudo code or otherwise, write an algorithm that will input Distance Travelled (km) and Fuel Used (litres) of cars, calculate their fuel economy and output their fuel economy.

Test data: 80, 10, 100, 5, 60, 2, 70, 5

First draw trace table write down column headings

Distance	Fuel	Fuel Economy

Setup in pseudo code using declaration of variable

Now Input Weight and height

Distance	Fuel	Fuel Economy

Input in pseudo code using test data

Now calculate the Fuel Economy using given formula

Distance	Fuel	Fuel Economy

Process in pseudo code using given formula

Now write down the above steps in pseudo code:

Q12.3) Write an algorithm, using pseudo code or flowchart only, which:

- inputs real numbers
- convert them into integer (whole) numbers

(You may use INT(X) in your answer e.g. Y = INT(3.8) gives the value Y = 3)

Test data: 80.9, 10.1, 100.8, 5.6

First draw trace table write down column headings

Number X	Integer Y	Output

Setup in pseudo code using declaration of variable

Now Input Number

Number X	Integer Y	Output

Input in pseudo code using test data

Now convert the real number into whole number using INT()

Number X	Integer Y	Output

Now write down the above steps in pseudo code:

Q12.4) Write an algorithm, using pseudo code or a program flowchart only, which:

- inputs the population and land area,
- calculates the population density (i.e. population/land area),

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

Q12.5) A system uses 5 digit numbers with an additional sixth digit used as a check digit.
(b) Each of the six digits in the number has a digit position. [Total=6]

6	5	4	3	2	1	←Digit position
a	b	c	d	e	f	
					Check digit	

digit in position 1 is the check digit i.e. f

The validity of the check digit is found using the following calculation:

- multiply each digit by its digit position (i.e. ax6, bx5, so on)
- add together the results of the multiplications
- divide the sum by 11
- If the remainder is ZERO then the number is valid

Write an algorithm, using pseudo code or flowchart only, which

- inputssix-digit barcodes in the form a, b, c, d, e and f
- re-calculates the check digit for each number and checks whether the input check digit (e) is correct

Q 12.6)Summer 2013

A small shop uses barcodes which represent 5 digits. The last digit is used as a check digit.

For example:

a b c d e
0 1 2 3 4

The check digit (e) is found by:

- multiplying the first and third digits (i.e. a and c) by 3
- multiplying the second and fourth digits (i.e. b and d) by 2
- adding these four results together to give a total
- dividing this total by 10
- remainder is check digit (e)

Write an algorithm, using pseudo code or flowchart only, which

- inputs five-digit barcodes in the form a, b, c, d, e
- re-calculates the check digit for each number and checks whether the input check digit (e) is correct

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

12.6

input 5 digits of the barcode (a, b, c, d and e) INSIDE a loop
find total value using barcode formula given
use MOD for finding remainder(e.g. 25 MOD 10 gives 5)
Output “Correct” or “Incorrect”

<u>example of suitable coding:</u>

```
INPUT a, b, c, d, e 1 mark
Total ← (a * 3) + (c * 3) + (b * 2) + (d * 2)
Remainder ← Total MOD 10
IF Remainder = e THEN OUTOUT “Correct” ELSE OUTPUT “Incorrect”
```

Q12.7) Define the term **algorithm**. **[2 marks]**

....

....

....

....

Q 12.7

A series of instructions//sequence of steps;
(Designed to) perform a particular task//solve a problem;

Q12.8)Followingpseudo code is used to find the largest value stored in an array.

```
1. Highest ← 0
2. FOR i ← 1 TO 10
3.     INPUT Number
4. IF Number > Highest THEN Highest ← Number
5. ENDFOR
6. PRINT Highest
```

(a) This pseudo code uses iteration. Give the line number on which iteration **starts.[1]**

....

(b) This pseudo code uses selection. Give the line number on which selection **starts.[1]**

....

(c) This pseudo code uses variable assignment. Give the line number in the function where variable assignment is **first** used. **[1]**

....

(d) The variable i in **Figure 1** only has scope between lines 2 and 5. Explain with reference to the variable i what scope means. **[1]**

....

12.8

a) *Line 2*
b) *Line 4*
c) *Line 1*
d) *The variable i can only be accessed/used/changed within those lines;*
The variable i is only defined within those lines;
Trying to access the variable i outside of those lines will not work;

Q12.9) This code is supposed to find out if a positive integer entered by a user is exactly divisible by the number 3.
Note: line numbers have been included and are not part of the code.

```
1INPUT n
2WHILE n ≥ 0
3       n ← n – 3
4ENDWHILE
5IF n = 0 THEN
6       OUTPUT 'is divisible by 3'
7ELSE
8       OUTPUT 'is not divisible by 3'
9ENDIF
```

The programmer realizes there is an error because a user input of 6 incorrectly outputs ‘is not divisible by 3'.

(a) In **Table** place a tick next to the type of error that the programmer has found.**[1]**

Type of error	Tick
Logical	
Runtime	
Syntax	

(b) State the line number of the code containing the mistake that causes this error to occur. **[1]**

....

(c) What change needs to be made to the line of code you have identified in your answer to**(b)** so that the program will work correctly? **[1]**

....

(d)What type of error could occur if the user enters the value eight? **[1]**

....

12.9

a Logical

b 4 or 7

c Any correct answer, examples include:
If the answer given for 9 (b) is 4 then
WHILE n > 0
WHILE n ≥ 1
WHILE n ≥ 3
If the answer given for 9 (a) (ii) is 7 then
IF n = -3 THEN

d Runtime error // Type error

Q12.10) The following pseudo code calculates the second hand price of different models of car.

The condition is an integer with a value between 1 and 4 where 1 is excellent and 4 is very bad.

```
INPUT Model, Condition, Age
cost ← 0
IF model = 'Daley' THEN
        cost ← 6000
ELSE IF model = 'Minty' THEN
        cost ← 4000
ELSE
        cost ← 2000
ENDIF

CASE condition OF
1: cost ← cost – 100
2: cost ← cost – 300
3: cost ← cost – 500
4: cost ← cost – 1000
ENDCASE
cost ← cost / age
PRINT cost
```

a) Tick the most appropriate data type of the variable cost. [1]

Data Type	**Tick onebox**
Boolean	
Character	
Real	
String	

b) Complete the trace table below showing the changes in the variable cost when the following values are input: **[4]**

"Tidy", 4, 2

Cost

12.10

a) *Real*

b) *1 mark for every correct row that appears in the correct sequence:*

cost
0
2000
1000
500

Q12.11) Write an algorithm, using pseudo code or flowchart only, which:

- inputs 1000 numbers
- outputs how many of these numbers were whole numbers (integers)

(You may use INT(x) in your answer, e.g. y = INT(3.8) gives the value y = 3)

.... [4]

(You may use INT(x) in your answer, e.g. y = INT(3.8) gives the value y = 3)

INPUT X	Y=INT(X)	Is X=Y?	CountINT
			0
3.8	3	No	
4	4	No	1
5	5	Yes	2
9.1	9	No	
7	7	Yes	3

Y=INT(X)
INT function
removes
fractional part

Initial value
CountINT← 0

CountINT←CountINT+ 1
Increment if X is an
integer

For integer
numbers X and y
will be equal

12.11

```
DECLARE Count, CountINT : Integer
DECLARE X, Y: Real
CountINT← 0
FOR Count ← 1 TO 1000
        PRINT “ Enter a number “
        INPUT X
        Y ← INT(X)
        IF X = Y THEN CountINT←CountINT + 1
NEXT Count
PRINT “ Number of integers = “ , CountINT
```

Q12.12) A restaurant table will have its data stored in its own booking record. Alessio decides to use an array of records.

Write **program code** to declare the array TableBookingsfor the 12 table records.

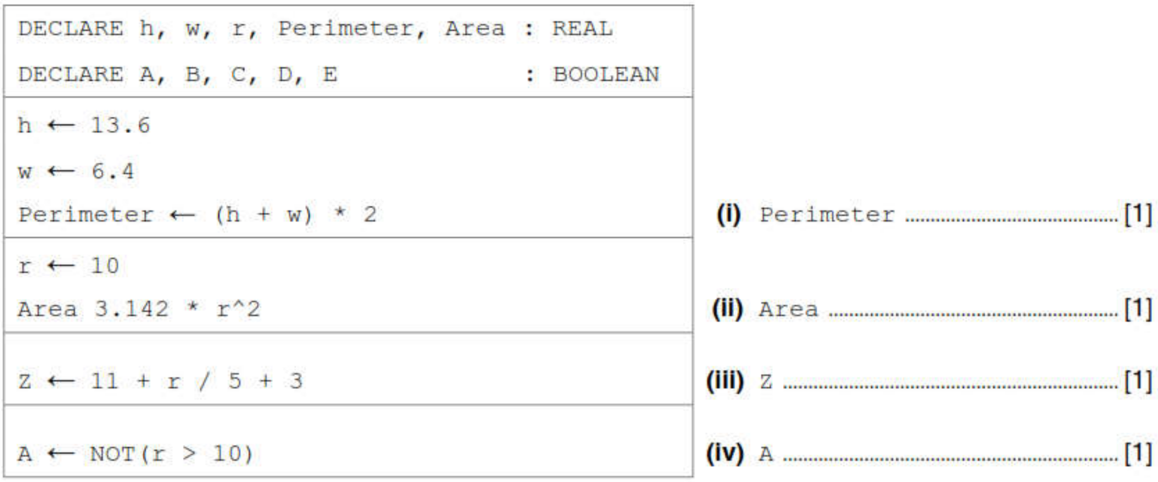
Programming language...

Code

....

....[1]

Q12.13) Computer programs have to evaluate expressions.
Study the sequence of pseudo code statements.
Write down the value assigned to each variable.



```
DECLARE h, w, r, Perimeter, Area : REAL
DECLARE A, B, C, D, E            : BOOLEAN

h ← 13.6
w ← 6.4
Perimeter ← (h + w) * 2

r ← 10
Area 3.142 * r^2

Z ← 11 + r / 5 + 3

A ← NOT(r > 10)
```

(i) Perimeter .... [1]

(ii) Area .... [1]

(iii) Z .... [1]

(iv) A .... [1]

Q 12.14) Computer programs have to evaluate expressions.
Study the sequence of pseudo code statements.
Give the value assigned to each variable.
The statement may generate an error. If so, write ERROR.
The & operator is used to concatenate strings.

```
DECLARE N1 : INTEGER
DECLARE N2 : INTEGER
DECLARE Answer : REAL
DECLARE Found : BOOLEAN
DECLARE IsValid : BOOLEAN
N1 ← 3
N2 ← 9
Answer ← (N1 + N2) / 6
Answer ← 3 * (N1 – 2) + N2 / 2
IsValid← (N1 > N2) AND (N2 = 9)
Found ← FALSE
IsValid← (N1 > N2 / 2) OR (Found = FALSE)
Answer "1034" & " + " & "65"
```

Give the value assigned to each variable

(i) Answer .... [1]

(ii) Answer .... [1]

(iii) IsValid... [1]

(iv) IsValid... [1]

(v) Answer .... [1]

Q12.16) Show what type of programming construct each statement represents.

Complete the table by putting a tick (✓) in the appropriate column for each item.

Item	**Statement**	**Selection**	**Iteration**	**Assignment**
1	MyScore = 65			
2	FOR IndexVal = 0 TO 99			
3	MyArray[3] = MID(MyString,3,2)			
4	IF MyScore>= 70 THEN			
5	ENDWHILE			
6	ELSE Message = "Error"			

Q12.17) Show what type of programming construct each statement represents.

Complete the table by putting a tick (✓) in the appropriate column for each item.

Item	**Statement**	**Selection**	**Iteration**	**Assignment**
1	`WHILE DegF> 37.5`			
2	`MyName = "Gordon"`			
3	`DegF = INT(DegF)`			
4	`ENDIF`			
5	`CASE OF MyFavourite`			
6	`UNTIL x = 5`			

Q12.18) A programmer uses an Integrated Development Environment (IDE) for all program development. Describe what is meant by an IDE.

....

....

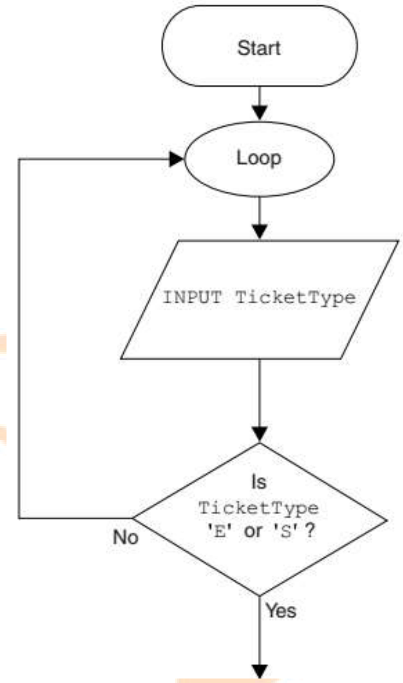
.... [2]

12.18

IDE is a (Single) software program

Features for:
program editor/writing/editing
translation // interpreter/compiler
testing program code // observe outputs   2 points to score

Q12.19)A program design is to be amended. The value input by the user for the ticket type is to be validated. Part of the amended flowchart is shown below.



Write **pseudo code** to use a pre-condition loop for this validation.

12.19

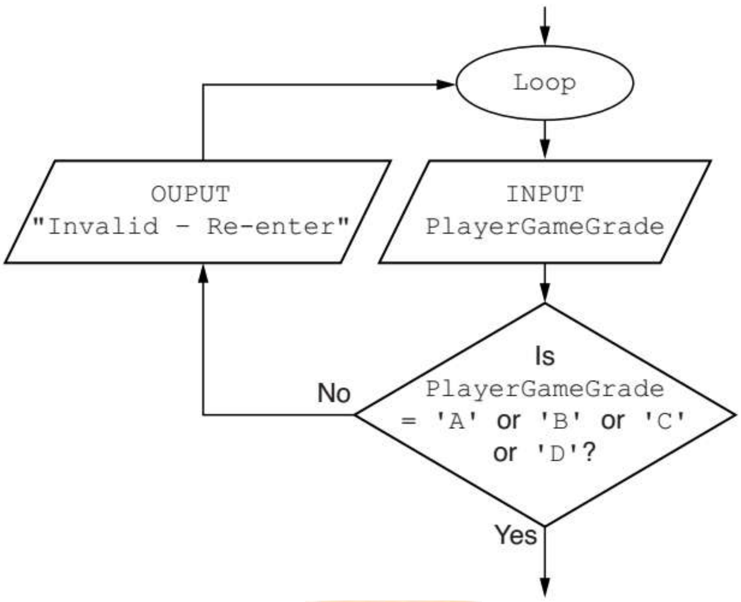
```
INPUT TicketType
WHILE TicketType<>"E" OR TicketType<>"S"
        INPUT TicketType
ENDWHILE
```

Or

```
INPUT TicketType
WHILE NOT (TicketType = 'E') OR (TicketType = 'S')
    INPUT TicketType
ENDWHILE
```

Q 12.20) The programmer amends the design to validate the value of player game grade that the user inputs.

The amended part of the flowchart is shown below.



Write the equivalent **pseudo code**using a pre-condition loop, for this part of the amended flowchart.

```
INPUT PlayerGameGrade
WHILE PlayerGameGrade<> 'A' OR PlayerGameGrade<> 'B' OR
        PlayerGameGrade<> 'C' OR PlayerGameGrade<> 'D')
        OUTPUT "Invalid – Re-enter"
        INPUT PlayerGameGrade
ENDWHILE
```

Or

```
INPUT PlayerGameGrade
WHILE NOT (PlayerGameGrade = 'A' OR PlayerGameGrade = 'B'
                   OR PlayerGameGrade = 'C' OR PlayerGameGrade = 'D')
        OUTPUT "Invalid – Re-enter"
        INPUT PlayerGameGrade
ENDWHILE
```

Q 12.21) A marathon runner records their time for a race in hours, minutes and seconds. An algorithm is shown below in structured English.

```
INPUT race time as hours, minutes and seconds
CALCULATE race time in seconds
STORE race time in seconds
OUTPUT race time in seconds
```

The identifier table needs to show the variables required to write a program for this algorithm. Complete the table.

Identifier	Data type	Description
RaceHours	INTEGER	The hours part of the race time.

Q12.22) Describe the purpose of each statement in this algorithm.

```
Count ← 0
WHILE Count<10 DO
        PRINT Count
```

```
                    Count ←Count + 1
          ENDWHILE
```

....

....

....

....

....[2]

Q12.23) Describe the purpose of each statement in this algorithm.

```
        Total ← 0
        INPUT Number
        REAPEAT
                Total ←Total + Number
                INPUT Number
        UNTIL Number < 0
        PRINT Total
```

....

....

....

....

....[2]

Q12.24) An algorithm is written that finds the mean average (i.e. the total of the numbers divided by how many numbers there are) of a set of 10 numbers stored in an array NumberArray.

```
        CONST Quantity = 10
        FOR Count =1 TO Quantity
                Total = Total + NumberArray(...................)
        NEXT Count
        Mean = ...........................................
        OUTPUT Mean
```

(a) Complete the algorithm by adding the missing statements. **[2]**

(b) Define the term constant, giving an example from the algorithm.

Definition ....

Example...**[3]**

(c) Identify the most appropriate data type for **Mean**. Justify your choice.

Data type ....

Justification ....**[2]**

(d) The algorithm uses iteration.

(i) Describe what is meant by iteration.

....**[2]**

(ii) Identify **two** forms of iteration that are **not** used in this algorithm.

1 ....

2 ....**[2]**

(e) The program is being extended to ask the user to enter numbers into the array. An algorithm is written to check that the input is valid.

```
REPEAT
    INPUT Number
UNTIL Number >= 0 AND Number <= 100
```

State **one** item of borderline data and **one** item of invalid data that can be input to test the algorithm works correctly.

Borderline ....

Invalid ....**[2]**

12.24

a)

1 mark for each pseudo code statement

*Total = Total + NumberArray(**Count**)*

*Mean = **Total/Quantity***

or

*Mean = **Total/Count***

or

*Mean = **Total/10***

***b)** 1 mark per bullet, max 2 for definition, 1 for example*

Definition:

•□A location in memory

•□A value/data that cannot be changed (whilst the program is running)

Example:

•□Quantity

c) 1 mark for data type, 1 for justification

Data type: Real/Float/Single/Double/Decimal

Justification: can be decimal/fractional/not a whole number

d i)1 mark per bullet, to max 2

•□A construct

•□Code is executed/run repeatedly//is looped

•□Until a condition is met/while a condition is true/a set number of times

d ii) •□While/do while

•□Repeat/ Repeat until/do until/ Until

e)1 mark for sensible borderline data, 1 mark for sensible invalid data.

•□Borderline – 0, 100

•□Invalid – number less than 0 (eg -1, -12) / number more than 100 (eg 101, 206) / non-numeric data (eg “test”, “#!%”)*

Q 12.25) Joseph is an author and programmer, and he needs to estimate how many pages his new book will have.

Each page has an average of 300 words. Each chapter starts with a chapter title page.

The number of pages is estimated by;

- dividing the number of words by 300
- ignoring the decimal part of the division
- adding the number of chapters to this total.

Joseph uses the algorithm below to estimate the number of pages, but his algorithm does not give the correct result.

```
01 INPUT numberOfWords
02 INPUT numberOfChapters
03 CONST wordsPerPage = 300
04 numberOfPages = RoundDown(numberOfWords / wordsPerPage)
05 numberOfPages = numberOfWords + numberOfChapters
06 OUTPUT numberOfPages
```

Joseph has used a RoundDownfunction to remove the decimal part of the division, e.g. RoundDown(6.2) would return 6, RoundDown(7.8) would return 7.

(a) State whether this algorithm uses selection, sequence or iteration.

.... **[1]**

(b) Line 03 defines a constant. Describe what is meant by a constant.

....

....

.... **[2]**

(c) There is an error in line 05 of the algorithm.

Write a corrected line of code to replace line 05.

.... **[1]**

(d) Identify the most appropriate data type for the following variable numberOfWords. Give areas on for your choice.

Data type ....

Reason ....**[2]**

(e) Joseph is changing his algorithm and needs to store the name and price of his book in new variables. State the most appropriate data type(s) for these variables.

Name ....

Price ....**[2]**

Joseph is using an Integrated Development Environment (IDE) to produce the program.

(f) One tool in an IDE that Joseph uses is a translator.

Describe **two** additional tools in an IDE that Joseph could use to help him produce his program.

Tool 1 name: ....

Tool 1 description: ....

....

Tool 2 name: ....

Tool 2 description: ....

....**[4]**

(g) Joseph's IDE allows him to use both a compiler and an interpreter.

Describe how Joseph could make use of a compiler and an interpreter when producing his program.

Compiler:...

....
....
....
Interpreter: ....
....
....
....[4]

12.25

a) Sequence
b) • A location in memory
*• The value/contents **cannot** be changed (whilst the program is running)*
***c)** numberOfPages = numberOfPages+numberOfChapters*
d) • Integer/Int
• It is a whole number/you can't have half a word
e) • String (name)
• Real/Single/Double/Currency/Float/(Decimal) (price)
f) 1 mark for identification, 1 for matching description
e.g.
• Error diagnostics/debugger
• …highlight errors/suggest changes
• Run-time environment
• …Lets you run/test the program
• Text editor
• …highlight key words
• …auto-indent
• …to type/edit source code
• …Auto-complete
• …highlight syntax errors
• Versioning tools
• …To allow for tracing back
• …To create new files with changes
• Stepping/breakpoints
• …Allow tracing of algorithms
g) Max 2 for compiler, 2 for interpreter
Compiler
• To convert to low-level in one go
• Create an executable//export the file
• To distribute the software
• Users will have no access to source code...
• ...so no-one can edit/steal/copy the code/program
• Use for error detection
Interpreter
• To convert to low-level line by line
• To test the program // to find errors
• stops running when it finds an error//shows the location of the error when found
• it is quicker (compared to compiler) to re-interpret than recompile

Q 12.26) A computer game has a stored number. The game gives the user 10 attempts to guess what the number is. If the user has got it correct, the game congratulates them and it ends. If the user has guessed it incorrectly, the game tells the user if the number is higher or lower than their guess.

Write an algorithm, using iteration, which:

- stores a number for the user to guess
- asks the user to guess the number
- outputs "congratulations" if the guess is correct and ends the game
- outputs if the user needs to guess lower, or higher
- allows the user 10 attempts to guess the number **[6]**

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

12.26

1 mark per bullet:

- *Storing a number for the user to guess*
- *Loops 10 times correctly*
- *Inputs the user's guess*
- *If correct, outputs congratulations **and** stops the loop / ends the game (any appropriate method of breaking out of loop)*
- *If the guess is greater than stored number, outputs lower (or similar)*
- *If the guess is lower than stored number, outputs higher (or similar)*

e.g. using while loop

```
num = 50 //(could be a random number)
x = 0
while x < 10
     input guess
     if guess == num then
          output "Congratulations"
          x = 10
     elseif guess >num then
          output "lower"
     else
          output "higher"
     endif
     x = x + 1
end while
```

e.g. example using for loop

```
num = 50 //(could be a random number)
for x = 1 to 10
     input guess
     if guess == num then
          output "Congratulations"
          end//(could be break / exit, or x = 10)
     elseif guess >num then
          output "lower"
     else
          output "higher"
     endif
next
```

Q 12.27) A memory game is played where:

- three players (A, B and C) have to choose a number between 0 and 100
- if the number has already been chosen, a message is displayed that says "taken"
- if the number has not already been chosen, the player's letter is placed next to it
- the quantity of numbers that have not yet been chosen is displayed.

The winner is the player who has chosen the most unique numbers by the end of the game. The numbers are stored in an array; numbers(). A number that has not yet been chosen is stored as an empty string "". The players are represented by "A", "B" and "C".

Figure shows an extract from the array:

Number:	0	1	2	3	4	...	...	99	100
Player:	A	C	B	A	B				

You have been asked to program part of the game.

Write an algorithm for player A's turn, which;

- takes as an input the number that player A chooses
- if it has not already been chosen, stores an "A" in that array element
- if it has already been chosen, outputs "taken"
- counts and outputs the quantity of numbers left that have not been chosen. **[6]**

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

12.27

1 mark per bullet

- *Taking the move as input*
- *Checking if array element input is free …*

o …Outputting if it is taken

- *Writing "A" to the correct array element*
- *Counting how many free space there are…*

o …Outputting the number of free spaces (if good attempt at counting free spaces)

e.g.

```
INPUT move
IF numbers(move) = "" then
```

```
        numbers(move) = "A"
ELSE
        output "taken"
ENDIF
free = 0
FOR x = 0 TO 100
        IF numbers(x) = "" then
                free = free + 1
        ENDIF
NEXT x
OUTPUT free
e.g.
INPUT move
IF numbers(move) = "" then
        numbers(move) = "A"
        numberFree = numberFree– 1
ELSE
        output "taken"
ENDIF

OUTPUT numberfree
```

Q 12.28) Charley is writing a program for music students. To make sure that there are no logic errors in the program, Charley uses a test plan.

(a) Describe what is meant by a logic error.

....

....

....

....**[2]**

(b) The program uses the letters in the following list to represent musical notes.

C D E F G A B

When the user inputs a letter from this list, the program outputs the next three notes in the list.If it gets to the end of the list, it starts again from the beginning, so the next note after B is C.Complete the test plan below by stating, for each input data, the expected outcome and areason for the test. **[6]**

Input Data	Expected outcome	Reason for test
C	....	....
A	....	....
H	....	....

12.28

a)•The error does not prevent program running…

•But it does not produce the expected output/it does not do what the programmer intended.

•A reasonable example

Input Data	***Expected outcome***	***Reason for test***
C	*D E F*	*checks the output is the next three letters in the list*
A	*BCD*	*checks the output goes back to the beginning of the list*
H	*Error message*	*Not a valid/existing note*

Q 12.29) Jim is writing a program to calculate the wages of workers in a teddy bear factory.

(a) Jim uses an Integrated Development Environment (IDE) to create the program.

Describe **two** tools in an IDE that can help Jim when creating the program.

1 ....

....

....

....

2 ....

....

....

....**[4]**

(b) Workers sometimes get a £50 bonus.
Here is the algorithm used to calculate whether a worker should get a bonus.

```
Limit = 200
INPUT WagesEarned
IF WagesEarned< Limit THEN
        Pay = WagesEarned
ELSE
        Pay = WagesEarned + 50
END IF
```

State the value of Pay after this code is executed for each of the following values of WagesEarned.

WagesEarned = 50 Pay = ...

WagesEarned = 200 Pay = ... **[2]**

(c) The wages earned by a worker is either £2 for every teddy bear they have made or £5 for every hour they have worked, whichever is larger.
Write an algorithm that:

- allows the user to input the number of teddy bears made and the number of hours worked
- calculates the wages for the number of teddy bears made
- calculates the wages for the number of hours worked
- outputs the larger of the two results.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... **[6]**

12.29

a) eg

•Editor

•Allows Jim to enter the program code

•Colour coding keywords

•Auto-completes code as you type.

•Compiler

•Transforms the written source code into machine code.

•Debugging tools

•Highlights errors in the code

•Suggests possible solutions.

b) •50

•250

c)eg

```
INPUT TeddyBears
INPUT Hours
PerTeddyBear = 2 * TeddyBears
PerHour = 5 * Hours
IF PerTeddyBear>PerHour THEN
        OUTPUT PerTeddyBear
ELSE
        OUTPUT PerHour
END IF
```

Award marks for:

•Inputting teddybears and hours

*•2 * number of teddy bears*

*•5 * hours*

•Comparing the two answers

•Outputting the piece rate if it is greater
•Outputting the hour rate if it is greater.

Q 12.30) A mail-order company buys dresses from America and France to sell in the UK. The company uses the following algorithm to convert sizes before printing them in its catalogue.
Half sizes are not possible (e.g. size 12.5).

```
INPUT Size
INPUT Origin
IF Origin = "America" THEN
    Size = Size + 2
ELSE
    IF Origin = "France" THEN
        Size = Size – 26
    END IF
END IF
PRINT Size
```

(a) The code uses the variables Origin and Size.
(i) Describe what is meant by a variable.
....
....
....
.... **[2]**

(ii) State the most appropriate data type for the variables Origin and Size, giving a reason for your choice.
Origin
Data type ....
Reason ....
....
Size
Data type ....
Reason ....
.... **[4]**

(b) The company sells the following dresses.

Dress A	**Dress B**	**Dress C**
Origin: France	Origin: America	Origin: UK
Size: 40	Size: 8	Size: 12

State the size which will be printed in the catalogue using the algorithm given.

Dress A ....

Dress B ....

Dress C ....**[3]**

12.30

a) (i) •A name/symbol which represents a value in a program

•... points to a memory location

•... and the value be changed (while the program is running)

a ii) ORIGIN

•Data type: String

•Reason: Consists of characters

Size

•Data type: Integer

•Reason: Consists of whole numbers

***b)** •Dress A: 14*

•Dress B: 10

•DressC: 1

Q 12.31) The program in a vending machine uses an array called Coins to store the value in pence of all the coins that have been entered in the current sale.

A maximum of 10 coins can be entered in each sale.

After each sale, the array is reset so that all values are 0.

(a) Here is an example of the contents of the array Coins during a sale.

10	100	20	50	5	0	0	0	0	0

In the example above, the value of Coins(1) is 10.

State the value of

Coins(4) ....

Coins(10) .... **[2]**

(b) An algorithm to reset the contents of the array Coins after each sale is shown below. This algorithm contains a logic error.

```
i = 1
REPEAT
    Coins(i) = 0
    i = i + 1
UNTIL i = 10
```

(i) State what is meant by a logic error.

....

.... **[1]**

(ii) Explain why the algorithm above contains a logic error.

....

....

....

.... [2]

(c) Write an algorithm to calculate the total value of the coins entered in the current sale using the contents of the array Coins.

....

....

....

....

....

....

....

.... [5]

12.31

a) •Coins(4) = 50

•Coins(10) = 0

(correct answers only)

(b) (i) •The program is written to do something other than what the programmer intended

(ii) •It will only reset the first 9 elements / will not reset the 10th element

•After setting Coins(9) = 0, i will become10...

•... and the loop will stop

•It should be UNTIL i > 10 / or other working correction

Example:

```
    I = 1
    Total = 0
    REPEAT
        total = total + Coins(i)
        i = i + 1
    UNTIL i>10 or Coins(i)=0
```

OR:

```
    total = 0
    FOR i = 1 to 10
        total = total + Coins(i)
    NEXT i
```

Award marks for:

•Initialising the total

•(Using a loop which) correctly starts from element 1

•... to element 10 / to the first 0 element

- *... each element is correctly added to the total*
- *... the iterator i (or equivalent) is correctly updated in the loop.*

Q 12.32) A syntax error can occur when writing a program.

(a) State what is meant by a syntax error, giving an example.

....

....

....

.... [2]

(b) Describe tools and facilities available in an integrated development environment (IDE) which can help the programmer to identify and correct syntax errors.

....

....

....

....

....

.... [4]

12.31

- *An error in the rules/grammar of the language*
- *Any suitable example*
- *Error messages/translator diagnostics*
- *... or on the fly while writing code*
- *Attempts to tell you what the error is*
- *And indicate where the error is/line numbers/underlines*
- *Editor...*
- *... allows you to enter the corrected code*
- *Produced when translating/by the compiler*

Q 12.32)A program contains the following code to calculate the circumference of a bicycle wheel, using the wheel size (diameter).

```
CONSTANT Pi = 3.14
INPUT WheelSize
Circumference = Pi * WheelSize
OUTPUT Circumference
```

(a) The code uses one constant and two variables.

(i) State the names of the constant and the variables.

Constant: ....

Variables: ....[2]

(ii) Explain **one** difference between a constant and a variable.

....

....

....

....**[2]**

(b) The data type of WheelSize is integer and the data type of Circumference is real number.Explain the difference between an integer and a real number.

....

....

....

.... **[2]**

12.32

a) (i) *•Constant: Pi*

•Variables: WheelSize, Circumference

(ii) •The value of a constant is set when the constant is declared

•The value of a variable is set while the program is running

•The value of a constant cannot be changed once the program is running/can only be set at design time

•A variable has no value at design time Marks in pairs

•An integer is a whole number

•A real number can include decimal fractions

Q12.33) A dog that is 5 years old is equivalent to a 42 year old human. Ashok is writing a program which

converts the age of a dog to the equivalent age for a human.

The program uses the following method:

- The user inputs age of the dog in years
- If the age is 2 or less, the human equivalent is 12 times the age
- If the age is more than 2, the human equivalent is 24 for the first 2 years, plus 6 for every additional year.

Write an algorithm to calculate and output the human equivalent of the age of a dog using the method described.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.......................**[5]**

12.33

Example:

```
BEGIN
    Input RealAge
    IF RealAge<= 2
        DogYears = RealAge * 12
    ELSE
        ExtraYears = RealAge – 2
        DogYears = 24 + ExtraYears * 6
    END IF
END
```

Award marks for an algorithm which:

•Allows real age to be input

•If age <=2, multiply real age by 12If age >2

•Works out extra years (real age – 2) ...

•... multiply by 6

•... adds 24 (for the first 2 years

Q 12.34) A syntax error can occur when writing a program.

(a) State what is meant by a syntax error, giving an example.

....

....

....

....**2]**

(b) Describe tools and facilities available in an integrated development environment (IDE) whichcanhelp the programmer to identify and correct syntax errors.

....

....

....

....**[4]**

12.34

(a)•An error in the rules/grammar of the language

•Any suitable example

(b)•Error messages/translator diagnostics

•Produced when translating/by the compiler

•... or on the fly while writing code

•Attempts to tell you what the error is

•And indicate where the error is/line numbers/underlines

•Editor...

•... allows you to enter the corrected code

Past paper questions on basic concepts of algorithm

Q 12.36 Specimen paper 2016 P2

2 Jatinder uses Internet banking.

This pseudo code checks her PIN.

```
c ← 0
INPUT PIN
x ← PIN
REPEAT
    x ← x/10
    c ← c + 1
UNTIL x < 1
IF c <> 5
    THEN
        PRINT "error in PIN entered"
    ELSE
        PRINT "PIN OK"
ENDIF
```

(a) What value of c and what message would be output if the following PINs were entered?

5 1 0 2 0 Value of c: ....

Message:..

5 1 2 0 Value of c:...

Message: .... [2]

(b) What type of validation check is being carried out here?

.... [1]

Q 12.37 Specimen paper 2016 P2

6 (a) Write an algorithm, using pseudo code or flowchart only, which:

- inputs three numbers
- outputs the largest of the three numbers

....

....

....

....

....

.... [3]

(b) Write an algorithm, using pseudo code or flowchart only, which:

• inputs 1000 numbers

• outputs how many of these numbers were whole numbers (integers)

(You may use INT(x) in your answer, e.g. y = INT(3.8) gives the value y = 3)

....

....

....

....

....

....

....

....

.... [4]

(c) Describe, with examples, two sets of test data you would use to test your algorithm.

1: ....

....

2: ....

.... [2]

Q 12.38 Summer 2015 P21& 23

4 Five data types and five data samples are shown below.

Draw a line to link each data type to the correct data sample. [4]

Data type	Data sample
Integer	'a'
Real	2
Char	2.0
String	True
Boolean	"Twelve"

Examiner Report Question 4

Nearly all candidates could link the data type of Boolean with the correct data sample. Some candidates confused Real and

Integer data types and/or String and Char data types.

Q 12.39 Summer 2015 P21& 23

5 Explain the difference between a variable and a constant in a program.

....

....

....

....[2]

Examiner Report Question 5
Well answered by many candidates.

Q 12.40 Summer 2015 P21& 23

6 Identify **three** different loop structures that you can use when writing pseudo code.

1..2

....

3.. [3]

Examiner Report Question 6

Most candidates could identify at least one loop structure. A common wrong answer was to incorrectly identify IF as part of a loop structure.

Q 12.41 Summer 2015 P22

4 Four programming concepts and four examples of programming code are shown below. Draw a line to link each programming concept to the correct example of programming code. [4]

Programming concept	**Example of programming code**
Counting	`Sum = Sum + Value[n]`
Repetition	`IF Value = 10 THEN PRINT 'X'`
Selection	`FOR Counter = 1 TO 10`
Totalling	`Amount = Amount + 1`
	`Sum = Num1 + Num2`

Examiner's comments on Question 4
Nearly all candidates could link the programming concept of selection with the correct example of programming code. Many candidates correctly linked at least three out of the four programming concepts. Due to an issue with this question, a discussion took place between the Principal Examiner and Assessment specialists to consider the impact on candidates in the light of answers seen. No candidates were disadvantaged and the full range of marks was seen.

Q 12.42 Summer 2015 P22

5 (a) Write an algorithm, using pseudo code and a `FOR … TO … NEXT` loop structure, to input 1000 numbers into an array.

....

....

....

....

....

....[2]

(b) Rewrite your algorithm using another loop structure.

....

....

....

....

....

....[4]

__Examiner's comments on Question 5__

(a) Most candidates attempted the loop structure, better candidates also showed the skill of being able to use the loop counter as the array index. Some candidates misread the question and incorrectly provided program code rather than pseudo code.

(b) Better candidates correctly used REPEAT … UNTIL or WHILE … DO … ENDWHILE structures.

The most challenging aspect was the correct management of the loop counter.

Q12.43 Winter 2015 P21 & 22

4 A routine checks the weight of melons to be sold in a supermarket. Melons weighing under0.5 kilograms are rejected and melons weighing over 2 kilograms are also rejected.

Give an example of each type of test data for this routine

Normal ....

Extreme ....

Abnormal ....[3]

__Examiners' Comments Question 4__

Most candidates could identify at least one correct example of test data. Examples of normal and abnormal test data were usually correct. Some candidates' examples of extreme test data were incorrect. A correct example of extreme test data would be 0.5 or 2.0.

Q12.44 Winter 2015 P21 & 22

5 Identify **two** different conditional statements that you can use when writing pseudo code.

1 ....

2 ....[2]

__Examiners' Comments Question 5__

Many candidates could identify IF as a conditional statement. Candidates with stronger responses throughout also identified CASE.

Q12.45 Winter 2015 P23

4 A routine checks the age and height of children who are allowed to enter a play area. The children must be less than 5 years of age and under 1 metre in height.

(a) The first set of test data used is age 3 and height 0.82 metres.

State what type of test data this is.

....

Give a reason for using this test data.

....

....[2]

(b) Provide **two** additional sets of test data. For each, give

- the type of each set of test data
- the reason why it is used

Each type of test data and reason for use must be different.

Set 1 ....

Type ....

Reason ....

....

Set 2 ....

Type ....

Reason ....

....[6]

Q 12.46 Summer 2016 P21 &P23

3 A program will be written to store information about members of a swimming club.

The following membership details will be recorded:

- Name
- Gender
- Status:
  - Senior
  - Junior
- Fee
- Team member (Yes or No)

(i) Choose a suitable data type for each of the membership details to be recorded. [5]

Membership details	**Data type**
Name	
Gender	
Status	
Fee	
Team member	

(ii) The swimming club has 50 members.

State the data structure that would be most suitable to use and give a reason for your choice.

Data structure ....

Reason ....

....[2]

Q 12.47 Summer 2016 P21 &P23

5 REPEAT ... UNTIL is one type of loop structure.

Identify and describe **two** other types of loop structure that you could use when writing pseudo code.

Loop structure 1: ....

Description: ....

....

Loop structure 2: ....

Description: ....

....[4]

Q 12.48 Summer 2016 P22

4 Four statement types and four examples are shown below.
Draw a line to connect each statement type to the correct example.

Statement type	Example
Assignment	`FOR X ← 1 TO 10`
Iteration	`READ X`
Input	`PRINT X`
Output	`X ← Y + Z`

[3]

Q 12.49 Summer 2016 P22

5 A programmer writes a program to store a patient’ s temperature every hour for a day. State the data structure that would be most suitable to use and give the reason for your choice.
Data structure ....
Reason ....
....[2]

Q 12.50 Summer 2016 P22

6 Identify two different selection statements that you can use when writing pseudo code.
1 ....
....
2 ....
....[2]

Q 12.51 Winter 2016 P21-23

4 Four validation checks and four descriptions are shown below.
Draw a line to link each validation check to the correct description. [3]

Validation check	**Description**
Presence check	Numbers between two given values are accepted
Range check	Data is of a particular specified type
Type check	Data contains an exact number of characters
Length check	Ensures that some data have been entered

Q 12.52 Winter 2016 P21-23

5 REPEAT ... UNTIL and WHILE ... DO ... ENDWHILE are two different loop structures you can use when writing pseudo code.
Explain, using examples, why you would choose to use each type of loop.
Example 1 ....
....
....
....
Reason for choice ....
....
....
Example 2 ....
....
....
....
Reason for choice ....
....
....[6]

Q 12.53 Winter 2016 P22

4 IF ... THEN ... ELSE ... ENDIF and CASE ... OF ... OTHERWISE ... ENDCASE
are two different conditional statements that you can use when writing pseudo code.
Explain, using examples, why you would choose to use each conditional statement.

Example 1 ....

....

....

....

....

....

Reason for choice ....

....

....

Example 2 ....

....

....

....

....

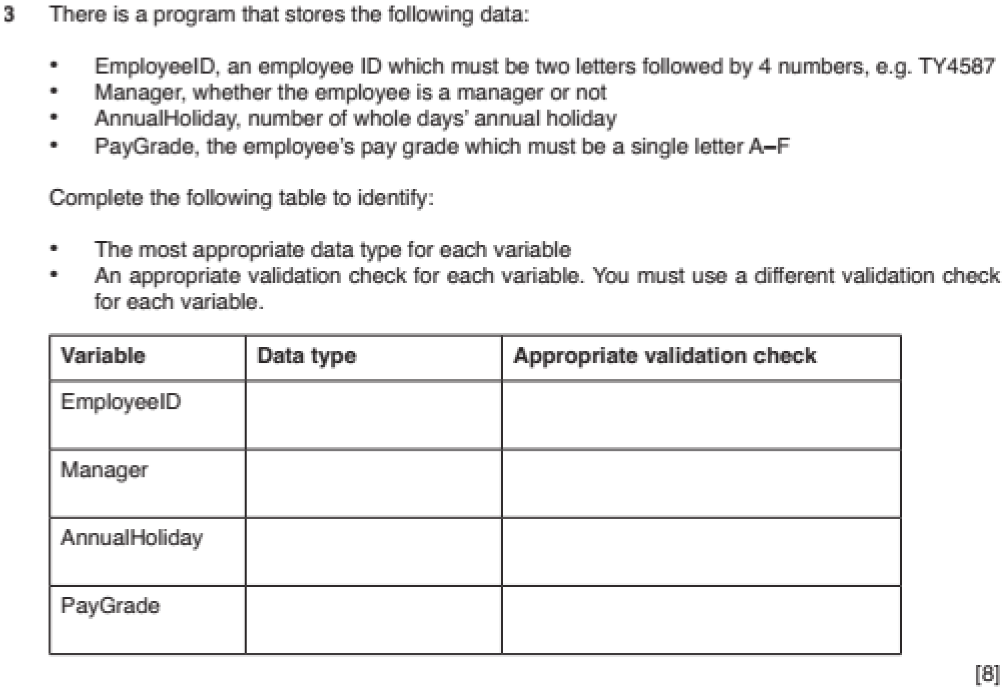
....

Reason for choice ....

....

.... [6]

Q 12.54 March 2017 P21 (India)



3 There is a program that stores the following data:

- EmployeeID, an employee ID which must be two letters followed by 4 numbers, e.g. TY4587
- Manager, whether the employee is a manager or not
- AnnualHoliday, number of whole days' annual holiday
- PayGrade, the employee's pay grade which must be a single letter A–F

Complete the following table to identify:

- The most appropriate data type for each variable
- An appropriate validation check for each variable. You must use a different validation check for each variable.

Variable	Data type	Appropriate validation check
EmployeeID		
Manager		
AnnualHoliday		
PayGrade		

[8]

Q 12.55 March 2017 P21 (India)

5 (a) Rewrite the following pseudocode algorithm using a WHILE … DO … ENDWHILE loop.

```
INPUT Num
FOR Counter ← 1 TO 12
    Num ← Num * Counter
    A[Counter] ← Num
NEXT
```

....

....

....

....

....

....

....

....

....

....[4]

(b) Explain the differences between a WHILE … DO … ENDWHILE and a REPEAT … UNTILloop.

....

....

....

....

....

....

....

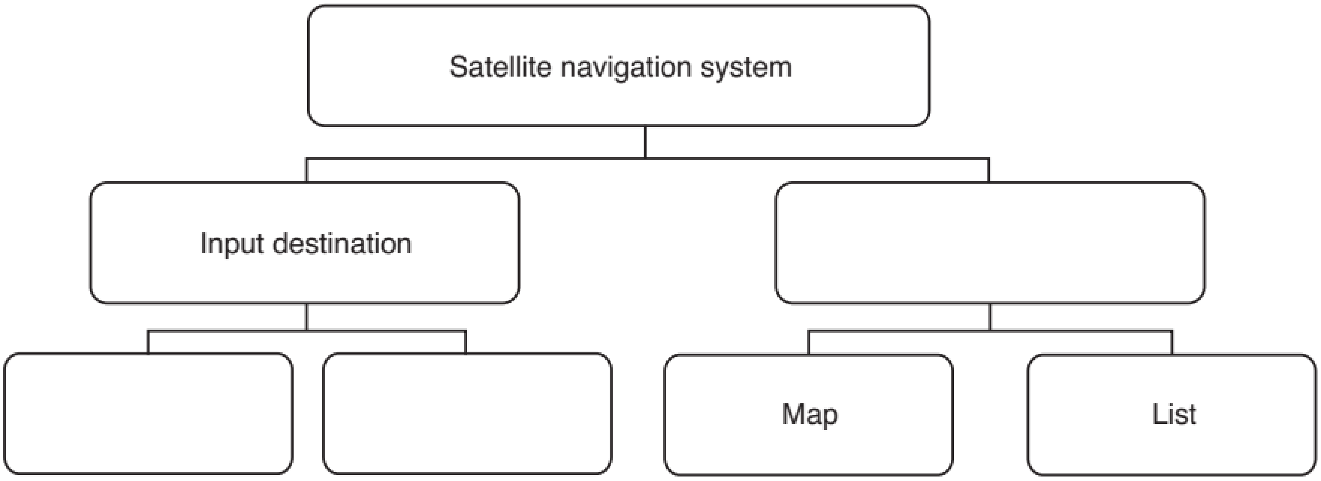
....[4]

Q 12.56 Summer 2017 P21

3 A satellite navigation system works using destination details entered by the user, either a new destination or chosen from previously saved destinations. The satellite navigation system will then output directions to the destination in the form of either a visual map or a list of directions.A satellite navigation system is an example of a computer system that is made up of sub-systems.

This structure diagram shows some of its sub-systems.

Complete the diagram by filling in the empty boxes. [2]



Q 12.57 Summer 2017 P21

4 For each of the **four** statements in the table, place a tick in the correct column to show whether it is an example of **validation** or **verification**. [4]

Statements	**Validation**	**Verification**
To automatically check the accuracy of a bar code		
To check if the data input is sensible		
To check if the data input matches the data that has been supplied		
To automatically check that all required data fields have beencompleted		

Q 12.58 Summer 2017 P21

5 (a) Describe the purpose of each statement in this algorithm.

```
FORI← 1 to 300
        INPUT Name[I]
NEXT I
```

....

....

....

....

....[2]

(b) Identify, using pseudocode, another loop structure that the algorithm in **part (a)** could have used.

....

....[1]

(c) Write an algorithm, using pseudocode, to input a number between 0 and 100 inclusive. The algorithm should prompt for the input and output an error message if the number is outside this range.

....

....

....

....

....

....

....

....

....

....[3]

Q 12.59 Summer 2017 P22

2 (a) Write an algorithm to input three different numbers, and then output the largest number.

Use **either** pseudocode **or** a flowchart.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[4]

(b) Give **two** sets of test data to use with your algorithm in **part (a)** and explain why you chose each set.

Test data set 1 ....

Reason ....

....

....

Test data set 2 ....

Reason ....

....

....[4]

Q 12.60 Summer 2017 P22

4 An algorithm has been written in pseudocode to input 100 numbers and print out the sum. A REPEAT … UNTIL loop has been used.

```
Count ←     0
Sum ←            0
REPEAT
    INPUT Number
    Sum ← Sum + Number
    Count ←Count + 1
UNTIL Count > 100
PRINT Sum
```

(a) Find the error in the pseudocode and suggest a correction.

Error..

Correction ....

....[2]

(b) Rewrite the correct algorithm using a more suitable loop structure.

....

....

....

....

....

....

....[3]

Q 12.61 Winter 2017 P21

3 (a) Explain the difference between a validation check and a verification check.

....

....

....[2]

(b) Describe, using an example, how data could be verified on data entry.

....

....

....[2]

(c) Explain what is meant by the term library routine.

....

....

....[2]

Q 12.61 Winter 2017 P21

4 (a) Fourpseudocode descriptions and **five** pseudocode statements are shown. Draw one line to link each pseudocode description to the correct pseudocode statement. Not all pseudocode statements will be used. [4]

Pseudocode description	Pseudocode statement
A loop that will iterate at least once.	FOR…TO…NEXT
A conditional statement to deal with many possible outcomes.	IF…THEN…ELSE…ENDIF
	WHILE…DO…ENDWHILE
A loop that will iterate a set number of times.	
	CASE…OF…OTHERWISE…ENDCASE
A conditional statement with different outcomes for true and false.	REPEAT…UNTIL

(b) Write an algorithm in pseudocode, using a single loop, to print 50 names that have been stored in an array.

....

....

....

....

....

....

....

....

....

....[3]

Q 12.62 Winter 2017 P22

2 Write an algorithm using **either** pseudocode **or** a flowchart, to:

- input a positive integer
- use this value to set up how many other numbers are to be input
- input these numbers
- calculate and output the total and the average of these numbers.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[6]

Q 12.63 Winter 2017 P22

3 The following diagram shows **four** data structures and **four** descriptions. [3]

Draw a line to connect each data structure to the correct description.

Data structure	Description
Constant	A collection of related data
Array	A value that can change whilst a program is running
Table	A value that never changes whilst a program is running
Variable	A series of elements of the same data type

Q 12.64 Winter 2017 P22

4 IF … THEN … ELSE … ENDIF is one type of conditional statement used when writing pseudocode.

Identify and describe **another** type of conditional statement that you could use when writing pseudocode. Give a reason why you would use this type of conditional statement.

Conditional statement ....

....

....

....

Description

....

....

....

....

Reason

....

.... [4]

Q 12.65 March 2018 P22 (India)

A program checks if the weight of a baby is at least 2 kilograms.
Give, with reasons, **two** different values of test data that could be used for the baby's weight.
Each reason must be different.
Value 1 ....
Reason ....
....
Value 2 ....
Reason ....
....[4]

Q 12.66 March 2018 P22 (India)

5 Explain the difference between the programming concepts of **sequence** and **selection**.
Include an example of a programming statement for each concept in your explanation.
....
....
....
....
....
....
....
....
....
....[4]

Comments on Question 5
*Candidates found the explanation of the difference between the programming concepts **sequence** and**selection** challenging, with few candidates identifying that programming statements in a sequence wereexecuted one after another whilst selection meant that the path through the program depends on the resultof a question. Candidates were more successful in providing suitable examples of programming statements.*
Common errors included confusing sequence or selection with iteration.

Q 12.67 Summer 2018 P21

2 (a) Write an algorithm to input 1000 numbers. Count how many numbers are positive and howmany numbers are zero. Then output the results. Use **either** pseudocode **or** a flowchart.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [6]

(b) Give one change you could make to your algorithm to ensure initial testing is moremanageable.

....

.... [1]

Q 12.68 Summer 2018 P21

4 A programmer has written a routine to check that prices are below $10.00. These values are usedas test data.

10.00 9.99 ten

Explain why each value was chosen.

10.00 ....

....

....

9.99 ....

....

....

ten ....

....

....[3]

Q 12.69 Summer 2018 P21

5 Explain the difference between the programming concepts of **counting** and **totalling**. Include an example of a programming statement for each concept in your explanation.

....

....

....

....

....

....

....

....

.... [4]

Q 12.70 Summer 2018 P22

2 (a) Draw a flowchart for an algorithm to input numbers. Reject any numbers that are negativeand count how many numbers are positive. When the number zero is input, the process endsand the count of positive numbers is output.

(b) Explain the changes you will make to your algorithm to also count the negative numbers.

....

....

....

....[2]

Q 12.71 Summer 2018 P22

4 A programmer has written a routine to store the name, email address and password of a contributorto a website's discussion group.

(a) The programmer has chosen to verify the name, email address and password.
Explain why verification was chosen and describe how the programmer would verify this data.

....

....

....

....

....

....

....

....[4]

(b) The programmer has also decided to validate the email address and the password.
Describe validation checks that could be used.

Email address..

....

....

Password ....

....

.... [2]

Q 12.72 Summer 2018 P22

5 A program checks that the weight of a basket of fruit is over 1.00 kilograms and under1.10 kilograms. Weights are recorded to an accuracy of two decimal places and any weight not inthis form has already been rejected.
Give **three** weights as test data and for each weight state a reason for choosing it. All your reasonsmust be different.

Weight 1..

Reason..

....

Weight 2..

Reason..

....

Weight 3..

Reason..

.... [3]

Marking Scheme

Q 12.36 Specimen paper 2016 P2

2 (a) 1 mark for value of c and message

51020:	value of c:	5	
	message:	PIN OK	(1 mark)
5120:	value of c:	4	
	message:	error in PIN entered	(1 mark)

(b) length check

Q 12.37 Specimen paper 2016 P2

6 (a) marking points:

the way to find and print the largest value a	1 mark
the way to find and print the largest value b	1 mark
the way to find and print the largest value c	1 mark

sample algorithm:

```
INPUT a, b, c
   IF a > b AND a > c THEN PRINT a              (1 mark)
      ELSE IF b > c THEN PRINT b                (1 mark)
         ELSE PRINT c                           (1 mark)
```

(b) marking points:

loop construct	1 mark
check if number is an *integer*	1 mark
counting the number of integers input	1 mark
output count value (outside the loop)	1 mark

sample algorithm:

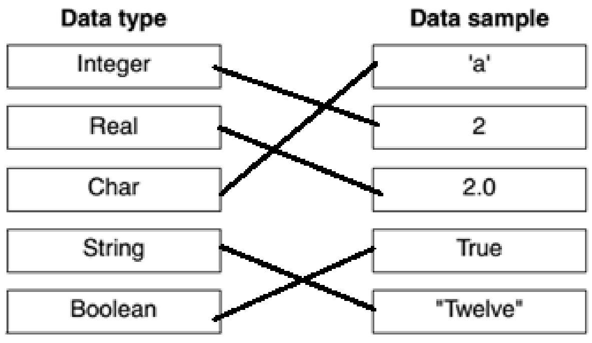
```
FOR x ← 1 TO 1000                                (1 mark)
   INPUT Number
      Difference ← INT(number) – Number          (1 mark)
      IF Difference = 0 THEN Total ← Total + 1   (1 mark)
NEXT x
PRINT total                                      (1 mark)
(NOTE: alternative to lines 3 and 4:
IF INT(Number) = Number THEN Total ← Total + 1   (2 marks) )
```

(c) Description of any **two** sets of test data. Many correct answers, these are exam

1000 whole numbers to ensure that loop works properly

900 whole numbers and 100 numbers with decimal places to ensure tha distinguishes correctly

Q 12.38 Summer 2015 P21& 23



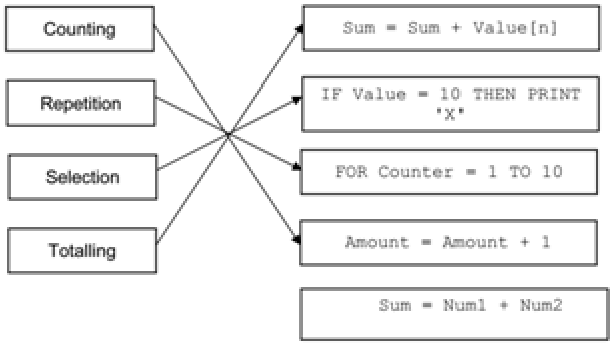
Q 12.39 Summer 2015 P21& 23

5 Any **two** points from
- a variable is used to store data that can change during the running of a program
- a constant is used to store data that will not be changed during the running of a program

Q 12.40 Summer 2015 P21& 23

6
- FOR (… TO … NEXT)
- REPEAT (… UNTIL)
- WHILE (… DO … ENDWHILE)

Q 12.41 Summer 2015 P22



Q 12.42 Summer 2015 P22

4 (a) (i) Normal

(ii) Acceptable data to test that the results are as expected.

(b) **One** mark for the data set, **one** mark for the type and **one** mark for the matching reason
There are many possible correct answers this is an example only.

Set 1	–	Age 4, height 0.9
Type	–	Boundary/Extreme
Reason	–	Data to test the validation that is just within the limits of acceptability
Set 2	–	Age 10, height 1.4
Type	–	Abnormal
Reason	–	Data that should be rejected and produce an error message

Q 12.46 Summer 2016 P21 &P23

(i) Name type – string
Gender type – char/string
Status type – char/string
Fee type – real
Team member type – Boolean [5]

(ii) Data Structure – several Arrays ……
………….Reason – to simplify programming/ make programs shorter/index can be used to identify the same member across the arrays etc. [2]

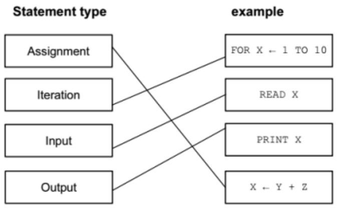
Q 12.47 Summer 2016 P21 &P23

5
- FOR (… TO … NEXT)…
- … a set number of iterations
- WHILE (… DO … ENDWHILE) …
- … used where the loop may never be executed/whilst a specified condition exists

Q 12.48 Summer 2016 P22

Page 5	Mark Scheme	Syllabus	Paper
	Cambridge IGCSE – May/June 2016	0478	22

4 1 mark for each correct line, maximum 3 (zero correct 0, one correct 1, two correct 2, three or four correct 3), each box must have only one connection.



Q 12.49 Summer 2016 P22

5 – data structure (one—dimensional) array ....
– .... reason to simplify programming/ make programs shorter, etc.

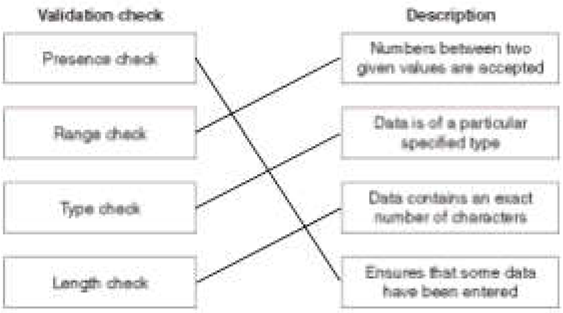
Q 12.50 Summer 2016 P22

6 – IF (… THEN … ELSE … ENDIF)

– CASE (… OF … OTHERWISE … ENDCASE)

Q 12.51 Winter 2016 P21-23

1 mark for each correct link up to maximum of 3 marks


Validation check
Description
Presence check
Range check
Type check
Length check
Numbers between two given values are accepted
Data is of a particular specified type
Data contains an exact number of characters
Ensures that some data have been entered

Q 12.52 Winter 2016 P21-23

5 For each example **1 mark** for **correct st**
1 mark for **reason.** There are many cor

```
REPEAT
..INPUT Number
  Total ← Total + Number
UNTIL Number = 0
```

– at least one repeat is required

```
WHILE Number <> -1 DO
..INPUT Number
   Total ← Total + Number
ENDWHILE
```

– the loop may never be executed

Q 12.53 Winter 2016 P22

4 For each example: **1 mark** for **correct structure, 1 mark** for **appropriate content, 1 mark** for **the reason.** There are many correct answers these are only examples

```
IF X > 0 AND X <= 10
  THEN PRINT 'In Range'
  ELSE PRINT 'Out of Range'
ENDIF
```

– e.g. checking a condition that may be complex//uses relational operators// checking for a range of values// only 2 options

```
CASE X OF
  1 : PRINT 'Option 1'
  2 : PRINT 'Option 2'
  3 : PRINT 'Option 3'
  OTHERWISE PRINT 'Incorrect choice'
ENDCASE
```

– e.g. checking for discrete/large number/more than 2 of values

Q 12.54 March 2017 P21 (India)

Variable	*Data Type*	*Appropriate Validation Check*
EmployeeID	String	Length Check / Presence Check / Format Check / Type check
Manager	Boolean	Type Check / Presence Check
AnnualHoliday	Integer	Type Check / Range Check / Presence Check
PayGrade	Char	Presence Check / Length Check / Type Check

Q 12.55 March 2017 P21 (India)

5(a)

– initialising counter outside the loop
– updating counter inside loop
– suitable exit value at start of loop
– correct use of WHILE … DO … ENDWHILE

Example:

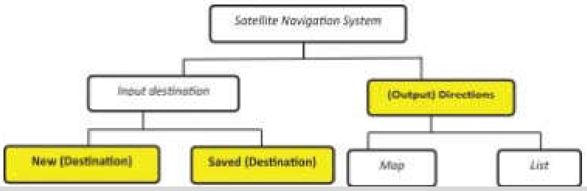
```
INPUT Num
Counter ← 1
WHILE Counter <= 12 DO
   Num ← Num * Counter
   A [Counter] ← Num
   Counter ← Counter + 1
ENDWHILE
```

5(b)

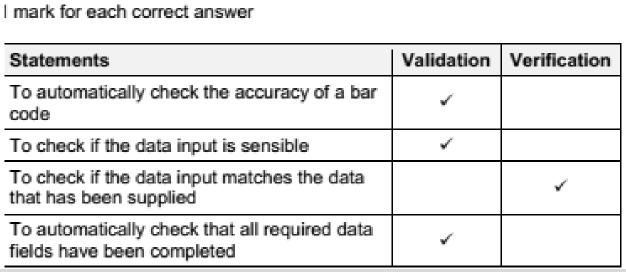
– WHILE has criteria check at start / pre-test
– may never run
– REPEAT UNTIL has criteria check at end / post-test
– will always run at least once

Q 12.56 Summer 2017 P21

Must match question.
2 marks for three suitable sub system names
1 mark for two suitable sub system names



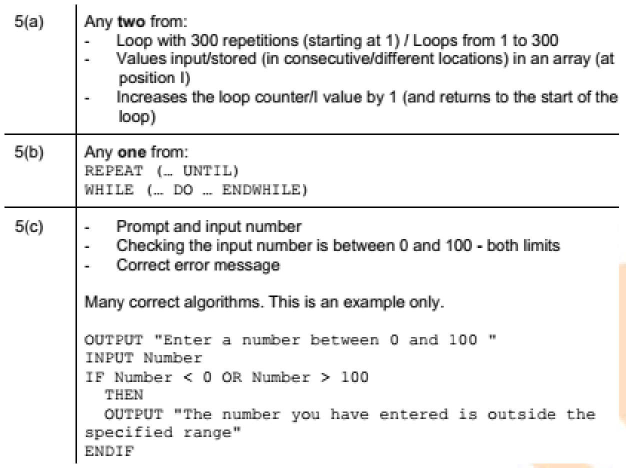
Q 12.57 Summer 2017 P21



I mark for each correct answer

Statements	Validation	Verification
To automatically check the accuracy of a bar code	✓	
To check if the data input is sensible	✓	
To check if the data input matches the data that has been supplied		✓
To automatically check that all required data fields have been completed	✓	

Q 12.58 Summer 2017 P21



5(a) Any **two** from:
- Loop with 300 repetitions (starting at 1) / Loops from 1 to 300
- Values input/stored (in consecutive/different locations) in an array (at position I)
- Increases the loop counter/I value by 1 (and returns to the start of the loop)

5(b) Any **one** from:

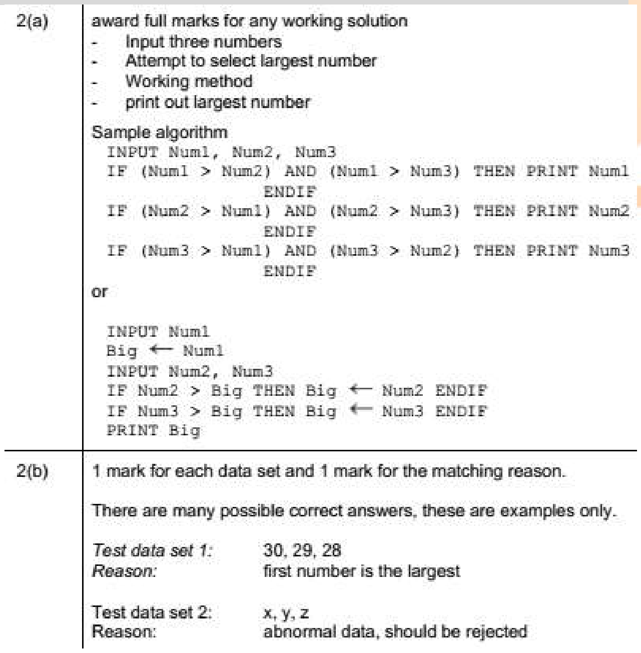
```
REPEAT (… UNTIL)
WHILE (… DO … ENDWHILE)
```

5(c)
- Prompt and input number
- Checking the input number is between 0 and 100 - both limits
- Correct error message

Many correct algorithms. This is an example only.

```
OUTPUT "Enter a number between 0 and 100 "
INPUT Number
IF Number < 0 OR Number > 100
  THEN
  OUTPUT "The number you have entered is outside the
specified range"
ENDIF
```

Q 12.59 Summer 2017 P22



2(a) award full marks for any working solution
- Input three numbers
- Attempt to select largest number
- Working method
- print out largest number

Sample algorithm

```
 INPUT Num1, Num2, Num3
 IF (Num1 > Num2) AND (Num1 > Num3) THEN PRINT Num1
                 ENDIF
 IF (Num2 > Num1) AND (Num2 > Num3) THEN PRINT Num2
                 ENDIF
 IF (Num3 > Num1) AND (Num3 > Num2) THEN PRINT Num3
                 ENDIF
```

or

```
 INPUT Num1
 Big ← Num1
 INPUT Num2, Num3
 IF Num2 > Big THEN Big ← Num2 ENDIF
 IF Num3 > Big THEN Big ← Num3 ENDIF
 PRINT Big
```

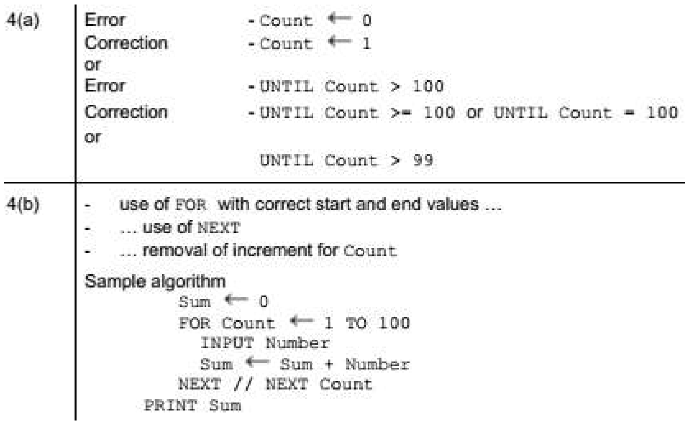
2(b) 1 mark for each data set and 1 mark for the matching reason.

There are many possible correct answers, these are examples only.

Test data set 1: 30, 29, 28
Reason: first number is the largest

Test data set 2: x, y, z
Reason: abnormal data, should be rejected

Q 12.60 Summer 2017 P22



4(a)

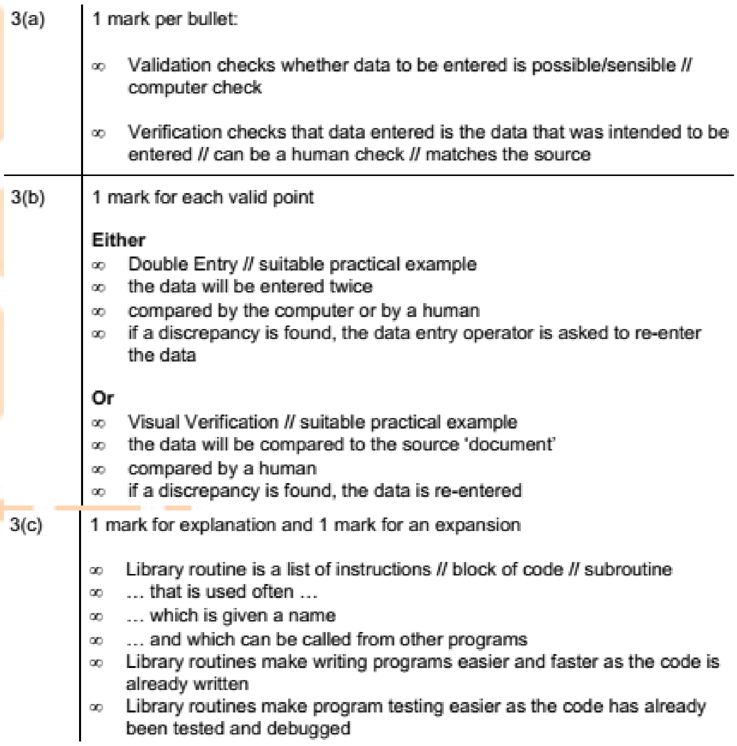
Error - `Count ← 0`
Correction - `Count ← 1`
or
Error - `UNTIL Count > 100`
Correction - `UNTIL Count >= 100` or `UNTIL Count = 100`
or
`UNTIL Count > 99`

4(b)
- use of `FOR` with correct start and end values …
- … use of `NEXT`
- … removal of increment for `Count`

Sample algorithm

```
        Sum ← 0
        FOR Count ← 1 TO 100
          INPUT Number
          Sum ← Sum + Number
        NEXT // NEXT Count
     PRINT Sum
```

Q 12.61 Winter 2017 P21



3(a) 1 mark per bullet:

- Validation checks whether data to be entered is possible/sensible // computer check
- Verification checks that data entered is the data that was intended to be entered // can be a human check // matches the source

3(b) 1 mark for each valid point

Either
- Double Entry // suitable practical example
- the data will be entered twice
- compared by the computer or by a human
- if a discrepancy is found, the data entry operator is asked to re-enter the data

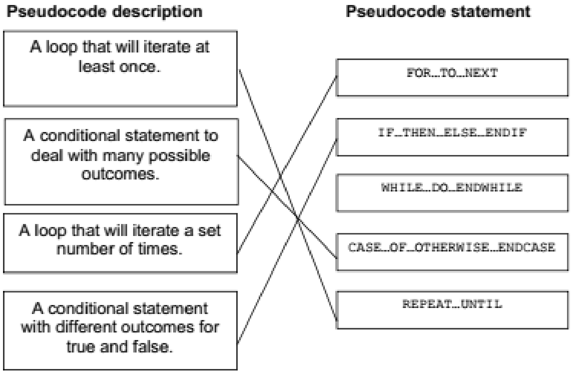
Or
- Visual Verification // suitable practical example
- the data will be compared to the source 'document'
- compared by a human
- if a discrepancy is found, the data is re-entered

3(c) 1 mark for explanation and 1 mark for an expansion

- Library routine is a list of instructions // block of code // subroutine
- … that is used often …
- … which is given a name
- … and which can be called from other programs
- Library routines make writing programs easier and faster as the code is already written
- Library routines make program testing easier as the code has already been tested and debugged

Q 12.61 Winter 2017 P21

4(a) 1 mark for each correct line


Pseudocode description
Pseudocode statement
A loop that will iterate at least once.
A conditional statement to deal with many possible outcomes.
A loop that will iterate a set number of times.
A conditional statement with different outcomes for true and false.
FOR…TO…NEXT
IF…THEN…ELSE…ENDIF
WHILE…DO…ENDWHILE
CASE…OF…OTHERWISE…ENDCASE
REPEAT…UNTIL

4(b) 1 mark per bullet:

- Appropriate loop controls
- Read from array
- Print from array (the last two points can be in one statement)

Note reading and printing **MUST** be within the same loop

Example algorithm:

```
Count ← 0
WHILE Count < 50 DO
   OUTPUT Name[Count]
   Count ← Count + 1
ENDWHILE
```

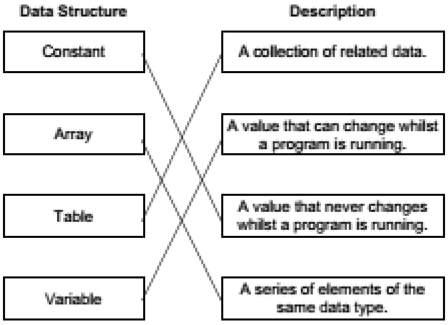
Q 12.62 Winter 2017 P22

any six from:
- initialise total (outside loop)
- Input number of numbers (outside loop with validation)
- Loop using input value
- Input number (inside loop)
- Update Total (inside loop)
- Calculate average
- Print average and total (outside loop)

Sample algorithm:

```
INPUT NumberCount
Total ← 0
FOR Count ← 1 TO NumberCount
  INPUT Number
  Total ← Total + Number
NEXT
Average ← Total/NumberCount
  PRINT Total, Average
```

Q 12.63 Winter 2017 P22


Data Structure
Description
Constant
Array
Table
Variable
A collection of related data.
A value that can change whilst a program is running.
A value that never changes whilst a program is running.
A series of elements of the same data type.

Q 12.64 Winter 2017 P22

Identification:

```
   CASE …
   … OF … OTHERWISE … (ENDCASE) or
   … OF … (OTHERWISE) … ENDCASE
```

Description:
- a statement that allows for multiple selections // not any of the above

Reason:
- to simplify pseudocode/ make pseudocode more understandable etc.

Q 12.65 March 2018 P22 (India)

1 mark for value and 1 mark for appropriate reason e.g.
Value 1 `2` (1) boundary should be accepted as weight OK (1)
Value 2 `two` (1) erroneous/abnormal should be rejected (1)

Q 12.66 March 2018 P22 (India)

2 marks for appropriate explanation, 1 mark example programming statements showing sequence, 1 mark example programming statement(s) showing selection e.g.

Sequence is the concept of one statement being executed after another(1) whereas selection decides which statement(s) are to be executed depending upon the result of a question (1)
sequence example (1)

```
PRINT X
PRINT Y
```

Selection example (1)

```
IF X > Y THEN PRINT X ELSE PRINT Y
```

Q 12.67 Summer 2018 P21

2(a) Any six from:
1 Initialisation of counters for positive numbers and zeros
2 Appropriate loop for 1000 iterations
3 Input number inside loop
4 Test for positive numbers
5 Update positive number counter
6 Test for zeros
7 Update zero counter
8 Output counters with appropriate messages outside loop

```
zero ← 0
posCount ← 0
FOR count ← 1 TO 1000
  INPUT number
  IF number > 0
    THEN posCount ← posCount + 1
  ENDIF
  IF number = 0
    THEN zero ← zero + 1
  ENDIF
NEXT
OUTPUT posCount, " positive numbers"
OUTPUT zero, " zeros"
```

2(b) Reduce the number of iterations to a manageable amount // Simulate the input

Q 12.68 Summer 2018 P21

One mark for each (max three)

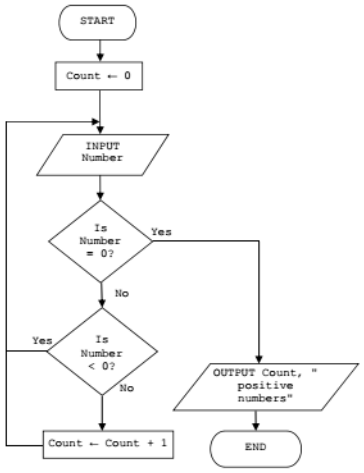
10.00	boundary/erroneous data // the price should be rejected // value is out of range
9.99	boundary/extreme/normal data // the prices should be accepted // value is within normal range
ten	erroneous/abnormal data // input should be rejected // value is wrong type

Q 12.69 Summer 2018 P21

There are many possible answers. e.g.:

Totalling is used to sum a list of numbers (1)
Counting is used to find how many numbers/items there are in a list. (1)
Totalling example (1) e.g. `Total ← Total + Number`
Counting example (1) e.g. `Counter ← Counter + 1`

Q 12.70 Summer 2018 P22

2(a)	**One** mark per correct pair of actions, process, Input/Output, Tests (apart from START and END) max 3 **One** mark complete Flowlines, **one** mark working flowlines, **one** mark correct use flowchart symbols  START Count ← 0 INPUT Number Is Number = 0? Yes No Is Number < 0? Yes No OUTPUT Count, " positive numbers" Count ← Count + 1 END
2(b)	Any **two** from: • Use another counter/variable • Update this counter/variable when the number is less than zero/count all numbers **and** subtract the positive numbers • Output this counter/variable at the end // Output both counters at the end

Q 12.71 Summer 2018 P22

4(a)	Max **4** in total Any **3** from: • To ensure no changes are made on input / accuracy of transcription • Because the details do not have fixed, values or lengths to validate • Because there is no clear set of rules that can be used for validation Any **3** from: • The programmer could ask the contributor to type in each detail twice … • … and then check that both values are equal • … If they are not equal then the input should be rejected • The programmer could ask the contributor to check the details on the screen … • … and confirm that they are correct / same as the original • … or change them
4(b)	**One** mark for email and **one** mark for password Email – check for @ / format check / no spaces /valid characters // presence check // length check (not more than 254 characters) // uniqueness check Password – length check / numbers and letters etc. // uniqueness check not been used before // presence check

Q 12.72 Summer 2018 P22

5	**One** mark per value and reason, max **3** Example 1.00 – boundary rejected//rejected (underweight) // out of range(1) 1.02 – normal // valid // accepted weight in range (1) 1.10 – abnormal // erroneous // invalid // rejected (overweight) (1)

Chapter 13

Errors in Pseudo code and Program

There are commonly three types of errors found in program codes:

A **syntax error** is a 'grammatical' error, in which a program statement does not follow the rules of the high-level language constructs. Due to syntax error program code can't be executed.

Logic error: an error in the logic of the solution that causes it not to behave as intended. Due to logical error a program is executed but doesn't produce required result.

Run-time error: an error that causes program execution to crash or freeze. E.g. divide-by-zero error.

Finding and correcting errors in pseudo code algorithm to Improve Efficiency

It is important to be able to identify errors and suggest corrections in a pseudo code algorithm.
If algorithm is correct but less efficient, students are asked to suggest improvements.
When task is changed, students are asked to modify pseudo code.
In loops following should points be considered:

- **Count-controlled loop** (FOR…TO…NEXT loop) should be used if number of repetition is given. For example input marks for 30 students,

```
FOR Count=1 TO 30
    INPUT marks
NEXT Count
```

- **Pre-conditioned loop** (WHILE…DO…ENDWHILE loop) should be used if loop is checked at the beginning and condition to continue the loop is given.For example to input only positive numbers, the numbers are validated at the time of input, when number is invalid, they are repeatedly input.

```
INPUT Number
WHILE Number<0 DO
        PRINT "Invalid number "
        INPUT Number
ENDWHILE
```

- **Post-conditioned loop** (REPEAT…UNTIL loop) should be used if loop is based upon a condition, but it has to be repeated at least once, and then condition to stop loop is checked. For example to input numbers, and calculate total until a rogue value like 0 is typed.

```
INPUT Number
REPEAT
    Total ←Total + Number
```

```
                INPUT Number
        UNTIL Number=0
```

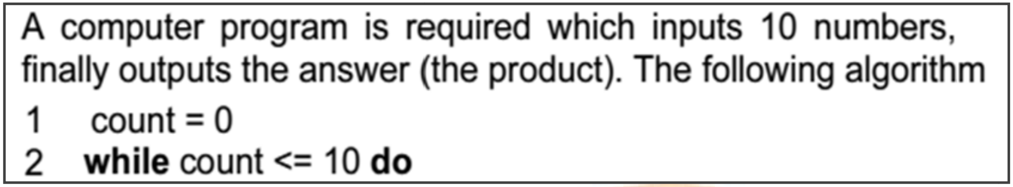
Common Errors in pseudo code:

There are 8 types of errors in pseudo code:

Error 1: Faulty initial or final value of loop counter

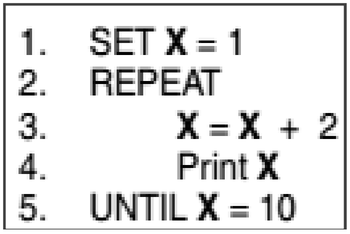
IF Count is initialized with 0 then Count< 'number of iteration' should be used in WHILE loop.

IF Count is initialized with 1 then Count<= 'number of iteration' should be used in WHILE loop.



A computer program is required which inputs 10 numbers, finally outputs the answer (the product). The following algorithm

```
1    count = 0
2   while count <= 10 do
```

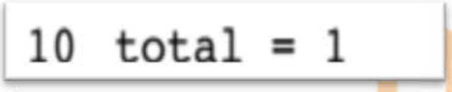


```
1.   SET X = 1
2.   REPEAT
3.          X = X + 2
4.          Print X
5.   UNTIL X = 10
```

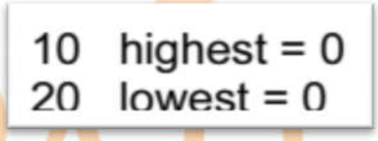
Error 2: Missing or Faulty initialization of a variable:

A variable must be initialized if it used in calculation without INPUT.

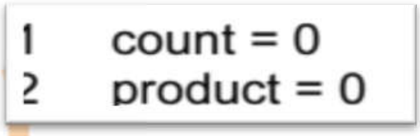
Total is initialized with 0, Product with 1, Highest with lowest possible value and Lowest with highest possible value.



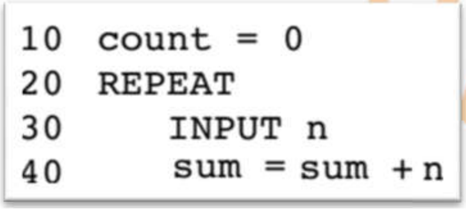
```
10  total = 1
```



```
10   highest = 0
20   lowest = 0
```



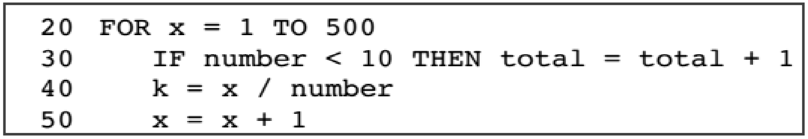
```
1      count = 0
2      product = 0
```



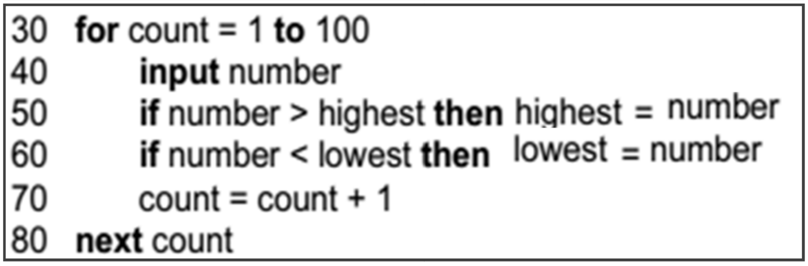
```
10  count = 0
20  REPEAT
30      INPUT n
40      sum = sum + n
```

Error 3: Increment in loop Counter in FOR…TO…NEXT loop.

FOR…TO…NEXT loop doesn't need increment in loop counter.



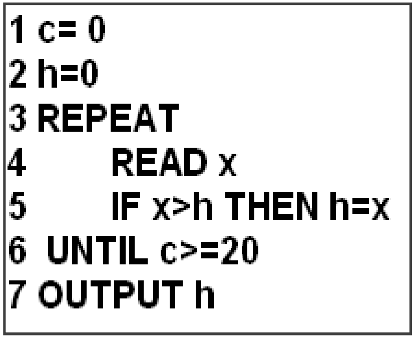
```
20  FOR x = 1 TO 500
30      IF number < 10 THEN total = total + 1
40      k = x / number
50      x = x + 1
```



```
30  for count = 1 to 100
40        input number
50        if number > highest then highest = number
60        if number < lowest then lowest = number
70        count = count + 1
80  next count
```

Error 4: Missing increment in loop Counter in REPEAT…UNTIL or WHILE…DO…ENDWHILE loop.

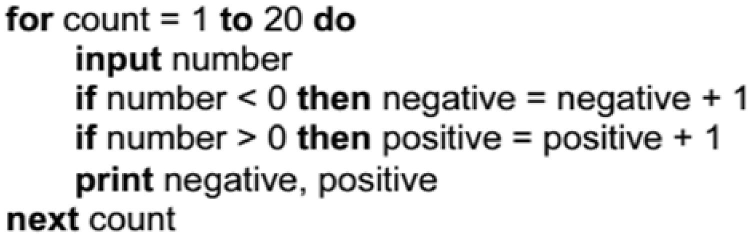
REPEAT…UNTIL loop and WHILE..DO..ENDWHILE loop needs increment in loop counter.



```
c= 0
h=0
REPEAT
    READ x
    IF x>h THEN h=x
 UNTIL c>=20
OUTPUT h
```

Error 5: Misplacing statement inside or outside of loop:

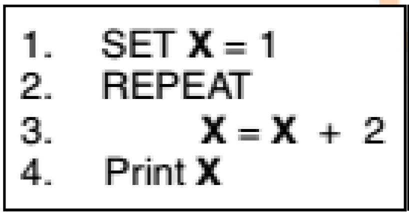
If Final output like greatest value or average is required it should be after loop.
IF running output is required it should be inside loop.



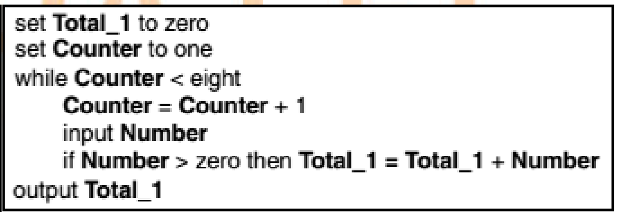
```
for count = 1 to 20 do
    input number
    if number < 0 then negative = negative + 1
    if number > 0 then positive = positive + 1
    print negative, positive
next count
```

Error 6: Missing ending keywords.

REPEAT…UNTIL or
WHILE…DO…ENDWHILE
FOR…TO…NEXT
IF…THEN…ENDIF
CASE ….OF…OTHERWISE….ENDCASE



```
SET X = 1
REPEAT
        X = X + 2
Print X
```

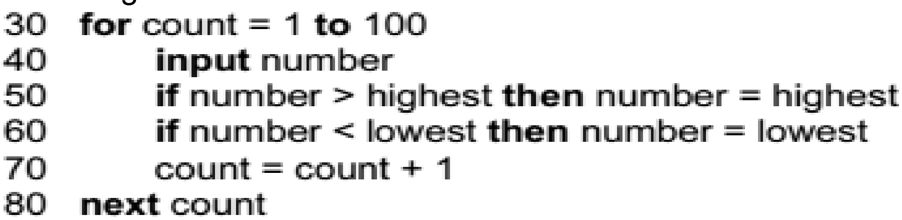


```
set Total_1 to zero
set Counter to one
while Counter < eight
    Counter = Counter + 1
    input Number
    if Number > zero then Total_1 = Total_1 + Number
output Total_1
```

Error 7: Assignment Error.

Values or vales of variable at right side should be assigned to variables and constants at left side.

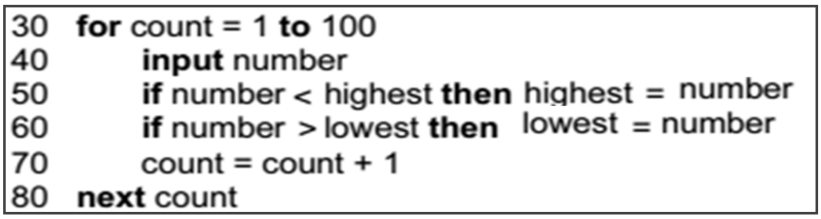
Number ← 58
Highest ← Number



```
30  for count = 1 to 100
40      input number
50      if number > highest then number = highest
60      if number < lowest then number = lowest
70      count = count + 1
80  next count
```

Error 8: Operator Error.
A common error in pseudo code is an improper operator.

IF number < Highest THEN Highest ←Number



```
30  for count = 1 to 100
40        input number
50        if number < highest then highest = number
60        if number > lowest then  lowest = number
70        count = count + 1
80  next count
```

Examination Questions

Q 13.1) Winter 2014 P13

The following pseudo code algorithm should:

- input up to 20 numbers
- stop if the sum of the input numbers **exceeds** 50
- output the final sum

```
10 count = 0
20 REPEAT
30 INPUT n
40 n + sum = sum
50 IF sum = 50 THEN count = 20
60 count = count + 1
70 UNTIL count = 20
80 OUTPUT n
```

There are **five** errors in this algorithm.

Locate these errors and suggest a correction.

error 1 ....

correction ....

....

error 2 ....

correction ....

....

error 3 ....

correction ....

....

error 4 ....

correction ....

....

error 5 ....

correction ....

....[5]

Q 13.2) Summer 2005

The following algorithm contains an error.

```
1. SET X = 1
2. REPEAT
3. X = X + 2
4. Print X
5. UNTIL X = 10
```

(a) Trace the algorithm and explain what the error is.

.... [2]

Q 13.3) Winter 2006

A computer program is required which inputs 10 numbers, multiplies them together and finally outputs the answer (the product). The following algorithm has been written to do this.

```
1       count  = 0
2       product  = 0
3       while  count<= 10 do
4       input number
5       product  = product  * number
6       count  = count  + 1
7       print product
8       endwhile
```

(a) There are three errors in the algorithm. Locate and describe these errors.

error 1 ....

correction ....

....

error 2 ....

correction ....

....

error 3 ....

correction ....

....

A while do loop has been used in the algorithm. State another type of loop that could have been used.

....

....

Q 13.4) Winter 2010

The following algorithm inputs 20 numbers and outputs how many numbers were positive (> 0) and how many numbers were negative (< 0).

```
1    negative = 1
2    positive = 1
3    for count = 1 to 20 do
4    input number
5    if number < 0 then negative = negative + 1
6    if number > 0 then positive = positive + 1
7    count = count + 1
8    print negative, positive
9    next count
```

There are three different errors in this algorithm.

Locate each error and give the reason why you think it is an error.

Error 1: ………………………………………………………………………………………..

Correction 1: ……………………………………………………………………………………

Error 2: ………………………………………………………………………………………..

Correction 2: ……………………………………………………………………………………

Error 3: ………………………………………………………………………………………..

Correction 3: ……………………………………………………………………………… [6]

Q 13.5) Summer 2011

Read the following section of code that inputs twenty (20) numbers and then outputs the largest number input.

```
1 h = 0
2 c = 0
3 REPEAT
4 READ x
5 IF x > h THEN x = h
6 c = c + 1
7 PRINT h
8 UNTIL c < 20
```

There are THREE errors in this code.

Locate these errors and suggest a corrected piece of code.

Error 1: ………………………………………………………………………………………..

Correction 1: ……………………………………………………………………………….

Error 2: ………………………………………………………………………………………..

Correction 2: ……………………………………………………………………………….

Error 3: ………………………………………………………………………………………..

Correction 3: [6]

Q 13.6) Winter 2013

A piece of pseudo code was written to input 1000 positive numbers and then output the highest and lowest numbers.

```
10 highest = 0
20 lowest = 0
30 for count = 1 to 100
40     input number
50     if number > highest then number = highest
60     if number < lowest then number = lowest
70     count = count + 1
80 next count
90 print highest, lowest
```

There are errors in the code.

Locate these errors and suggest a correction.

error 1 ....

correction: ....

....

error 2: ....

correction: ....

....

error 3: ....

correction: ....

....

error 4: ....

correction ....

....

Q 13.7) Winter 2014 P12

The following section of a pseudo code algorithm should:

- input 500 numbers
- generate a ratio called **k**
- output each value of **k**
- output how many numbers were larger than 10

```
10 total = 1
20 FOR x = 1 TO 500
30      IF number < 10 THEN total = total + 1
40      k = x / number
50      x = x + 1
60      OUTPUT k
70 NEXT x
80 OUTPUT x
```

(a) There are **five** errors in the above code.

Locate these errors and suggest a correction.

error 1 ....

correction: ....

....

error 2: ....

correction: ....

....

error 3: ....

correction: ....

....

error 4: ....

correction ....

....

error 5: ....

correction ....

....

(b) The corrected algorithm was converted to a computer program and run. However, after several numbers were input, the program stopped and an error message was generated, showing that there was a further error at line 40 (k = x / number).

State what could cause this error to occur.

....

....

Suggest a change to line 40 to overcome this problem...

....[2]

Questions from Past Papers

13.8 Summer 2015 P21& 23

2 Read this section of program code that should input 10 positive numbers and then output the smallest number input.

```
1 Small = 0
2 Counter = 0
3 REPEAT
4 INPUT Num
5 IF Num< Small THEN Num = Small
6 Counter = Counter + 1
7 PRINT Small
8 UNTIL Counter < 10
```

There are **four** errors in this code.

Locate these errors and suggest a corrected piece of code for each error.

1 ....

....

2 ....

....

3 ....

....

4 ....

.... [4]

Examiner Report Question 2

Most candidates located at least one error and suggested a suitable piece of corrected code. The error on line 8 was often identified, with better candidates providing a working correction.

13.9 Summer 2015 P22

2 Read this section of program code that should input 30 positive numbers and then output the largest number input.

```
1 Large = 9999
2 Counter = 0
3 WHILE Counter > 30
4 DO
5 INPUT Num
6 IF Num< Large THEN Large = Num
7 Counter = Counter - 1
8 ENDWHILE
9 PRINT Large
```

There are **four** errors in this code.

Locate these errors and suggest a corrected piece of code for each error.

1

....

....

2...

....

3...

....

4

....

....[4]

Examiner's comments on Question 2

Most candidates located at least one error and suggested a suitable piece of corrected code. The error on line seven was the one identified and corrected by nearly all candidates. The error on line 3 was often identified, with better candidates providing a working correction.

13.10 Winter 2015 P21 & 22

2 Read this section of program code that should input 50 numbers and then output the average.

```
1 Total = 0
2 For Counter = 1 TO 50
3 INPUT Num
4 Total = Total + 1
5 Counter = Counter + 1
6 Average = Total/Counter
7 NEXT Counter
8 PRINT Average
```

There are **four** errors in this code. Locate these errors and suggest code corrections to remove each error.

1. ....

....

2 ....

....

3 ....

....

4 ....

....[4]

Examiners' Comments Question 2

Many candidates located at least one error and suggested a suitable piece of corrected code. The errors on lines 4 and 5 were frequently identified, with stronger responses providing a working correction. The question asked the candidates to identify and correct each error; a few candidates either identified the error or corrected the error, but both actions were required to gain each mark.

13.11 Summer 20162210,0478 P22

2 Read this section of program code that inputs 10 positive numbers and then outputs the total.

```
Total = 0
Counter = 0
REPEAT
INPUT Num
Total = Total + Num
PRINT Total
Counter = Counter + 1
UNTIL Counter = 10
```

This code works, but it is inefficient.

(i) Suggest **three** improvements that could be made.

1 ....

....

2 ....

....

3 ....

....[3]

(ii) Rewrite the program code with your improvements.

....

....

....

....

....

....

....

....

....[3]

13.12 Specimen paper 20162210,0478 P2

4 Read this section of program code that inputs twenty (20) numbers and then outputs the largest number input.

```
1 h = 0
2 c = 0
3 REPEAT
4 READ x
5 IF x > h THEN x = h
6 c = c + 1
7 PRINT h
8 UNTIL c < 20
```

There are three errors in this code.

Locate these errors and suggest a corrected piece of code.

1: ....

2: ....

3: .... [3]

13.13 Winter 2016 P21-23

2 Read this section of program code that:

- inputs 10 numbers
- checks whether each number is within a specified range
- totals the numbers within the range and outside the range

```
1 InRange = 0
2 OutRange = 1000
3 FOR Count = 1 TO 10
4 INPUT Num
5 IF Num> 10 AND Num< 20 THEN InRange = InRange + 1
6 ELSE OutRange = OutRange - 1
7 Count = Count + 1
8 NEXT X
9 PRINT InRange, OutRange
```

(a) There are four errors in this code.

Locate these errors and suggest a correction to remove each error.

Error 1 ....

Correction ....

....

Error 2 ....

Correction ....

....

Error 3 ....

Correction ....

....

Error 4 ....

Correction ....

....[4]

(b) Decide, with reasons, whether the numbers 10 and 20 are within or outside the range. [4]

Number	**Withinrange (✓)**	**Outside range (✓)**	**Reason**
10			....
20			....

13.14 Winter 2016 P22

2 Read this section of program code that inputs positive numbers, discards any negative numbers and then outputs the average. An input of zero ends the process.

```
1 Total = 0
2 Counter = 100
3 REPEAT
4 REPEAT
5 INPUT Num
6 UNTIL Num< 0
7 Total = Total + 1
8 Counter = Counter + Num
9 UNTIL Num = 0
10 Average = Total / (Counter - 1)
11 Print Average
```

There are four errors in this code.

Locate these errors and suggest a correction to remove each error.

Error 1 ....

Correction ....

....

Error 2 ....

Correction ....

....

Error 3 ....

Correction ....

....

Error 4 ....

Correction ....

.... [8]

13.15 March 2017 P21 (India)

2 Read this section of code that inputs the ages of people entering an event. The input sequence is ended by inputting a negative value for age. The code outputs the number of people at the event over the age of 18.

```
Num18 = 0
INPUT Age
WHILE Age >= 0 DO
    IF Age >= 18 THEN
        Num18 = Num18 + Age
    ENDIF
ENDWHILE
PRINT Num18 – Age
```

There are four errors in this code.

Locate these errors and suggest code correction to remove each error.

Error 1 ....

Correction..

....

Error 2 ....

Correction..

....

Error 3 ....

Correction..

....

Error 4 ....

Correction..

.... [4]

13.16 Summer 2017 P21

2 This section of program code asks for 50 numbers to be entered. The total and average of the numbers are calculated.

```
1 Total = 0
2 Counter = 50
3 PRINT ′When prompted, enter 50 numbers, one at a time′
4 REPEAT
5       PRINT ′Enter a number′
6       INPUT Number
7       Total + Number = Total
8       Number = Number + 1
9 UNTIL Counter = 50
10 Average = Number * Counter
11 PRINT ′The average of the numbers you entered is ′, Average
```

There are **four** errors in this code.

State the line number for each error and write the correct code for that line.

Error 1 Line number ....

Correct code ....

Error 2 Line number ....

Correct code ....

Error 3 Line number ....

Correct code ....

Error 4 Line number ....

Correct code ....[4]

13.17 Summer 2017 P22

4 An algorithm has been written in pseudo code to input 100 numbers and print out the sum.

A REPEAT … UNTIL loop has been used.

```
Count ←      0
Sum ←              0
REPEAT
    INPUT Number
    Sum ← Sum + Number
    Count ←Count + 1
UNTIL Count > 100
PRINT Sum
```

(a) Find the error in the pseudo code and suggest a correction.

Error...

Correction ....

....[2]

(b) Rewrite the correct algorithm using a more suitable loop structure.

....

....

....

....

....

....

.... [3]

13.18 Winter 2017 P21

2 This section of program code asks for 80 numbers between 100 and 1000 to be entered. It checks that the numbers are in the correct range, and stores them in an array. It counts how many of the numbers are larger than 500 and then outputs the result when the program is finished.

```
1 Count = 0
2 FOR Index = 1 TO 80
3       INPUT 'Enter a number between 100 and 1000', Number
4       WHILE Number = 99 AND Number = 1001
5             INPUT 'This is incorrect, please try again', Number
6       ENDWHILE
7       Num[80] = Number
8       IF Number > 500 THEN Count = Count + 1
9 UNTIL Index = 80
10 PRINT Index
11 PRINT ' numbers were larger than 500'
```

There are **four** lines of code that contain errors.

State the line number for each error and write the correct code for that line.

Error 1 Line Number ....

Correct Code ....

Error 2 Line Number ....

Correct Code ....

Error 3 Line Number ....

Correct Code ....

Error 4 Line Number ....

Correct Code ....[4]

13.19 March 2018 P22 (India)

2 An algorithm has been written in pseudo code to input some numbers and print out any numbers that are greater than or equal to 100. The number 999 stops the algorithm.

```
INPUT Number
WHILE NUMBERS <> 999 DO
    IF Number > 100 THEN PRINT Number ENDIF
ENDWHILE
PRINT Number
```

(a) Find the **four** errors in the pseudo code and suggest corrections.

Error 1 ....

Correction ....

....

Error 2 ....

Correction ....

....

Error 3 ....

Correction ....

....

Error 4 ....

Correction ....

.... [4]

(b) Show, using pseudo code, how you would change the corrected algorithm to print out anynumbers between 100 and 200 inclusive.

....

....

....

....

....

....

....

....[2]

Comments on Question 2

(a) *Most candidates correctly identified one or two errors. A few candidates showed good understanding of the pseudo code by correctly identifying the problem with the variable name and the need to add INPUT Number before ENDWHILE. A common error was to suggest that the WHILE condition was incorrect.*

(b) *Some candidates realised that as well as introducing an upper bound, there was a change required to the value of the lower bound of the selection test, as the number 100 would now be included.*

Finding output from pseudo code

13.20 Winter 2001

This algorithm grades candidates on marks out of ten.

```
1 input a Mark
2   case Mark of
3       0, 1, 2, 3 : Grade = Fail
4       4, 5 : Grade = Pass
5       6, 7 : Grade = Merit
6       8, 9, 10 : Grade = Distinction
7       otherwise Mark = -1
8 endcase
9 if Mark = -1 then
10      print 'Not a valid mark'
11 else output Grade, 'Grade'
```

(a) Dry run the algorithm for each of the following data and complete the table. [3]

INPUT	OUTPUT
0	
5	
99	

(b) Write down two instructions which could be inserted between lines 1 and 2 to allow the algorithm to deal with marks out of 100. [2]

13.21Specimen 2016

Jatinder uses Internet banking. This pseudo code checks her PIN.

```
c ← 0
INPUT PIN
x ← PIN
REPEAT
        x ← x/10
        c ← c + 1
UNTIL x < 1
IF c <> 5THEN
        PRINT "error in PIN entered"
ELSE
        PRINT "PIN OK"
ENDIF
```

C	PIN	X	OUTPUT
0			
	51020	51020	
1		5102	
2		510.2	
3		51.02	
4		5.102	
5		0.5102	
			PIN OK

(a) What value of c and what message would be output if the following PINs were entered?

5 1 0 2 0 Value of c: 5

Message: PIN OK

5 1 2 0 Value of c: 4 .

Message: "error in PIN entered" [2]

(b) What type of validation check is being carried out here?

Length check (Checks number of digits in PIN. [1]

13.22Winter 2002

Read this algorithm.

```
set Total_1 to zero
set Total_2 to zero
set Counter to one
while Counter < eight
        Counter = Counter + 1
        input Number
        if Number > zero then Total_1 = Total_1 + Number
        if Number < zero then Total_2 = Total_2 + Number
endwhile
output Total_1
output Total_2
```

(a) Write down the output if the following set of numbers are input. 4, 1, -3, 2, -5, 0, 6

.... [2]

(b) Modify the algorithm so that it will accept any number of numbers, the input is terminated by a rogue value and the output is the Total of all the numbers input except the rogue value.

.... [4]

13.23 Summer 2003

Read this algorithm.

```
input A, B
if A > B then
        T = A
        A = B
        B = T
endif
output A, B
```

(a) Write down the output if the following two numbers are input:

41, 38 ....[1]

(b) Explain the purpose of the variable T .... [1]

(c) Explain why an algorithm is written as a subroutine (procedure) and stored in a program library. ....[2]

13.24 Winter 2003

The following algorithm inputs air speeds (which must be in multiples of 100) and outputs a suitable message.

```
1 input a speed
2 whole = speed/100
3 case whole of
4       0,1,2 : result = slow
5       3, 4, 5, 6 : result = normal
8       7, 8, 9 : result = high
7       otherwise whole = -1
8 endcase
9 if whole = -1 then
10      output “abnormal reading”
11 else output result, “speed”
```

Dry run the above algorithm for the following Input data and complete the Output column in the table: [3]

Input	Output
150	
400	
800	

State what would be happen if line 2 had been missed out of the algorithm?

....

.... [2]

13.25

(b) Write an algorithm which uses a While..Do..Endwhile loop and outputs the numbers 2, 4, 6 and 8[3]

Marking Scheme

Q 13.1) Winter 2014 P13

error: line 10: sum not initialised
correction: sum = 0

error: line 40: incorrect formula for sum
correction: sum = sum + n

error: line 50: incorrect IF statement
correction: IF sum > 50 THEN ……

error: lines 50 and 60: value of count causes a problem e.g. loop never ending
correction: **either** count = 19 on line 50
or count = count + 1 between lines 30 and 40
or any other correct solution

error: line 80: output of n is incorrect
correction **output** sum or **print** sum

Q 13.2) Summer 2005

(a) Award **1** mark each for trace and reason:

trace – 3,5,7,9,11……

reason – x is odd/loop does not terminate/goes on forever

(b) Award **1** mark for the following stages:

initialise
loop
use of x = x + 2
output of x

Q 13.3) Winter 2006

9 **(a)** error 1: product = 0 on line 2
should use product = 1

error 2: loop control, count <= 10 on line 3
should use count < 10 or alternatively alter count value on line 1 to count = 1

error 3: print value of product inside loop on line 7
output should come after the endwhile statement [3]

(b) Accept either of the following loop controls:

```
repeat                                  for count = 1 to 10
                     OR
until count = 10                        next count
(accept repeat

       until count ≥ 11

 if line 1 changed to count = 1)
```

Q 13.4) Winter 2010

1 mark for each error and **1 mark** for reason why it is an error

- line 1/negative=1 and/or line 2/positive=1
- negative and/or positive should be set to zero
- line 7/count=count+1
- don't need a count within a **for …. to next** loop
- replace loop with a **repeat…until** loop
- line 8/**print** negative, positive or line 9/**next** count
- outputs should come after the **next count** statement

Q 13.5) Summer 2011

(a) 1 mark for each error identified + suggested correction

line 5: this should read ***if x > h then h = x***

line 7: ***print h*** should come <u>after the end of the repeat loop</u>

line 8: this should read ***until c = 20*** or ***until c >= 20*** or ***until c > 19***

(b) Any **two** from:
- close to English
- one statement is equal to many low-level language statements
- portable
- easy to edit/debug/update
- problem oriented
- needs converting to machine code before execution

(c) Any **one** from:
- interpreter – runs line by line and locates errors as it runs
- compiler – converts whole program into object code/gives complete list of errors

Q 13.6) Winter 2013

(8) 1 mark for error + 1 mark for suggested correction to error (max of **FOUR errors**)

description of possible error	suggested correction to error
line 20 lowest = 0	lowest = 100 (or even bigger value)
line 30 loop count is 1 to 100	count should be 1 to 1000 e.g. **for** count = 1 **to** 1000
line 50 number = highest	formula is reversed e.g. should be: highest = number
line 60 number = lowest	formula is reversed e.g. should be: lowest = number
line 70 count = count + 1 addition of count in a **for … to** loop	remove line 70 from coding

Q 13.7) Winter 2014 P12

(a) 1 mark for each error and suggested correction (accept description or example of corrected pseudocode).

error: line 10: total = 1
correction: totals should be set to zero; total = 0

error: line 30: … number < 10 …
correction: check should be made if number > 10; … number > 10 …

error: no input inside loop
correction: **input** number

error: line 50: x = x + 1
correction: **for … to** loops don't need a counter; remove line 50 altogether

error: line 80: **output** x
correction: output should be total value; **output** total [5]

(b) division by zero error (or similar description of error produced when dividing by 0)

add an error trap after input of number
e.g. 40 **if** number = 0 **then** k = 0 **else** k = x/number [2]

Questions from Past Papers

13.8 Summer 2015 P21& 23

1 mark for each error identified + suggested correction
Line 1 or `Small = 0`: this should read **`Small = 999`**
line 5 or `IF…`: this should read **`IF Num < Small THEN Small = Num`**
line 8 or `UNTIL`: this should read **`UNTIL Counter = 10`** or
`UNTIL Counter > = 10` or
`UNTIL Counter > 9`
line 7 or `PRINT…`: **`PRINT Small`** should come after the end of the repeat loop
or
line 8 or `UNTIL`: this should come before line 7

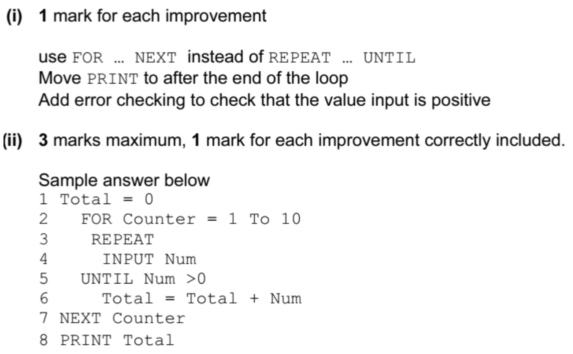
13.9 Summer 2015 P22

1 mark for each error identified + suggested correction
Line 1 or `Large =9999`: this should read **`Large = 0`**
Line 3 or `WHILE`: this should read **`WHILE Counter < 30`**
line 6 or `IF`: this should read **`IF Num > Large THEN Large = Num`**
line 7 or `Counter =…`: this should read **`Counter = Counter + 1`**

13.10 Winter 2015 P21 & 22

One mark for each error identified + suggested correction
line 4 or (`Total =) Total + 1`: this should read (`Total =) Total + Num`
line 5 or `Counter = Counter + 1`: delete this line
line 6 or (`Average = )Total / Counter`: swap lines 6 and 7
line 6 or (`Average = )Total / Counter` : this should read (`Average =) Total / 50`

13.11 Summer 2016 P22



(i) **1** mark for each improvement

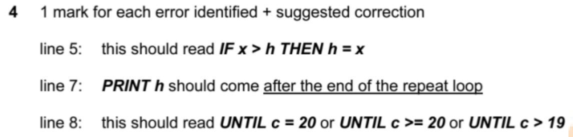
use `FOR … NEXT` instead of `REPEAT … UNTIL`
Move `PRINT` to after the end of the loop
Add error checking to check that the value input is positive

(ii) **3** marks maximum, **1** mark for each improvement correctly included.

Sample answer below

```
Total = 0
  FOR Counter = 1 To 10
   REPEAT
     INPUT Num
  UNTIL Num >0
     Total = Total + Num
NEXT Counter
PRINT Total
```

13.12 Specimen paper 2016 P2



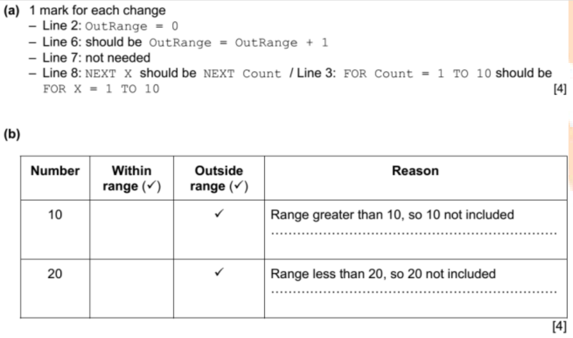
4 1 mark for each error identified + suggested correction

line 5: this should read ***IF x > h THEN h = x***

line 7: ***PRINT h*** should come after the end of the repeat loop

line 8: this should read ***UNTIL c = 20*** or ***UNTIL c >= 20*** or ***UNTIL c > 19***

13.13 Winter 2016 P21-23



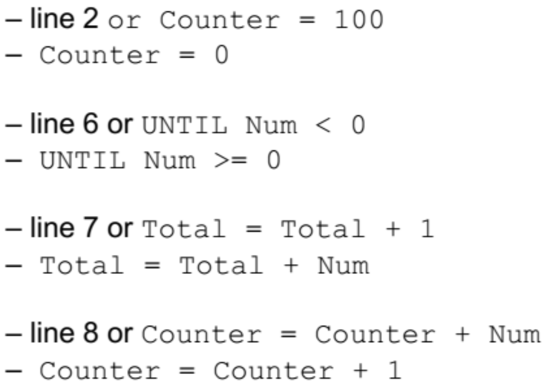
(a) 1 mark for each change
- Line 2: `OutRange = 0`
- Line 6: should be `OutRange = OutRange + 1`
- Line 7: not needed
- Line 8: `NEXT X` should be `NEXT Count` / Line 3: `FOR Count = 1 TO 10` should be `FOR X = 1 TO 10` [4]

(b)

Number	Within range (✓)	Outside range (✓)	Reason
10		✓	Range greater than 10, so 10 not included
20		✓	Range less than 20, so 20 not included

[4]

13.14 Winter 2016 P22



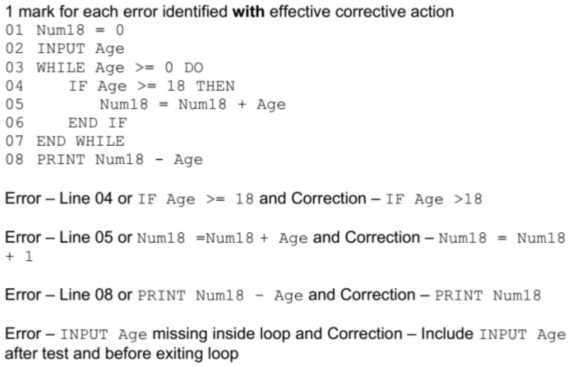
– line 2 or `Counter = 100`
– `Counter = 0`

– line 6 or `UNTIL Num < 0`
– `UNTIL Num >= 0`

– line 7 or `Total = Total + 1`
– `Total = Total + Num`

– line 8 or `Counter = Counter + Num`
– `Counter = Counter + 1`

13.15 March 2017 P21 (India)



1 mark for each error identified **with** effective corrective action

```
01 Num18 = 0
02 INPUT Age
03 WHILE Age >= 0 DO
04     IF Age >= 18 THEN
05         Num18 = Num18 + Age
06     END IF
07 END WHILE
08 PRINT Num18 - Age
```

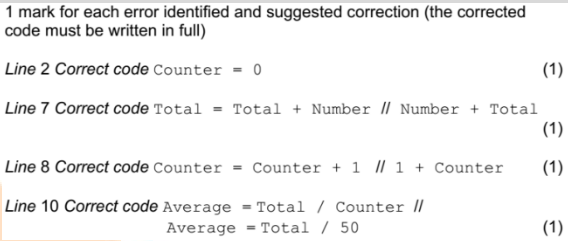
Error – Line 04 or `IF Age >= 18` and Correction – `IF Age >18`

Error – Line 05 or `Num18 =Num18 + Age` and Correction – `Num18 = Num18 + 1`

Error – Line 08 or `PRINT Num18 – Age` and Correction – `PRINT Num18`

Error – `INPUT Age` missing inside loop and Correction – Include `INPUT Age` after test and before exiting loop

13.16 Summer 2017 P21



1 mark for each error identified and suggested correction (the corrected code must be written in full)

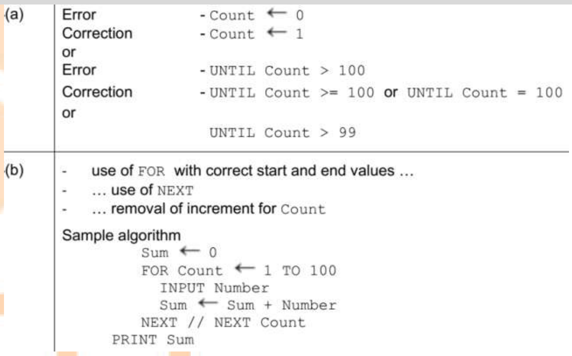
Line 2 *Correct code* `Counter = 0` (1)

Line 7 *Correct code* `Total = Total + Number` // `Number + Total` (1)

Line 8 *Correct code* `Counter = Counter + 1` // `1 + Counter` (1)

Line 10 *Correct code* `Average = Total / Counter` //
`Average = Total / 50` (1)

13.17 Summer 2017 P22



(a)

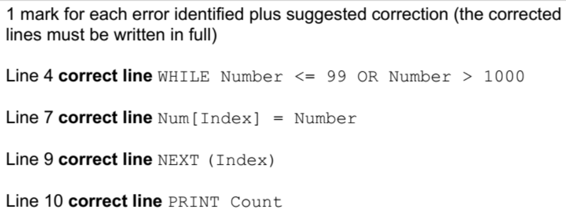
Error	- `Count ← 0`
Correction	- `Count ← 1`
or	
Error	- `UNTIL Count > 100`
Correction	- `UNTIL Count >= 100` or `UNTIL Count = 100`
or	`UNTIL Count > 99`

(b)
- use of `FOR` with correct start and end values …
- … use of `NEXT`
- … removal of increment for `Count`

Sample algorithm

```
     Sum ← 0
     FOR Count ← 1 TO 100
       INPUT Number
       Sum ← Sum + Number
     NEXT // NEXT Count
PRINT Sum
```

13.18 Winter 2017 P21



1 mark for each error identified plus suggested correction (the corrected lines must be written in full)

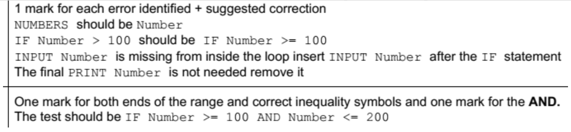
Line 4 **correct line** `WHILE Number <= 99 OR Number > 1000`

Line 7 **correct line** `Num[Index] = Number`

Line 9 **correct line** `NEXT (Index)`

Line 10 **correct line** `PRINT Count`

13.19 March 2018 P22 (India)



1 mark for each error identified + suggested correction
`NUMBERS` should be `Number`
`IF Number > 100` should be `IF Number >= 100`
`INPUT Number` is missing from inside the loop insert `INPUT Number` after the `IF` statement
The final `PRINT Number` is not needed remove it

One mark for both ends of the range and correct inequality symbols and one mark for the **AND.**
The test should be `IF Number >= 100 AND Number <= 200`

Finding output from pseudo code

13.20 Winter 2001

(a) **One mark per correct *output*:**

Input	Output
0	***Fail Grade***
5	***Pass Grade***
99	***Not a valid mark***

(NOTE: accept the words FAIL, PASS without the word GRADE. If the word GRADE precedes the words FAIL, PASS still accept the answer. The letters "P" and "F" on their own = 0 marks)

[3]

(b) For example (1 mark per line up to the maximum):
mark = mark/10
mark = INT(mark)

mark = mark DIV10 is worth 2 marks on its own [2]

13.21 Specimen 2016

51020:	value of c:	5
	message:	PIN OK
5120:	value of c:	4
	message:	error in PIN entered

length check

13.22 Winter 2002

13.23 Summer 2003

(a) 38, 41 or A =38, B = 41 or B = 41, A =38

(b) temporary store to facilitate the swap

(c) Award **one** mark for each:
less program code
less memory
used in other programs
parameter passing
e.g. sorting, finding the maximum

13.24 Winter 2003

(a)

150	abnormal reading
400	normal speed
800	high speed

(ignore word "speed" in answer) [3]

(b) any **two** points from:
only data 0 to 9 would register
all other data would give "abnormal reading" message/incorrect response
variable **whole** would not exist
thus **whole** would be zero OR algorithm would crash/fail [2]

13.25

Chapter 14

Database

2.3 Database

- define a single-table database from given data storage requirements
- choose and specify suitable data types
- choose a suitable primary key for a database table
- perform a query-by-example from given search criteria

A database is a collection of information organized to provide efficient retrieval. The collected information could be in any number of formats (electronic, printed, graphic, audio, statistical, combinations). There are physical (paper/print) and electronic databases.

A database could be as simple as an alphabetical arrangement of names in an address book or as complex as a database that provides information in a combination of formats.

Examples:

- phone book
- address book
- Census Bureau data

Database Management System (DBMS)

Database management system is a mechanism for manipulating data with high level command. It hides low level details such are as how data are obtained.

Database management system also has ability to search record by queries and to create reports and view data.

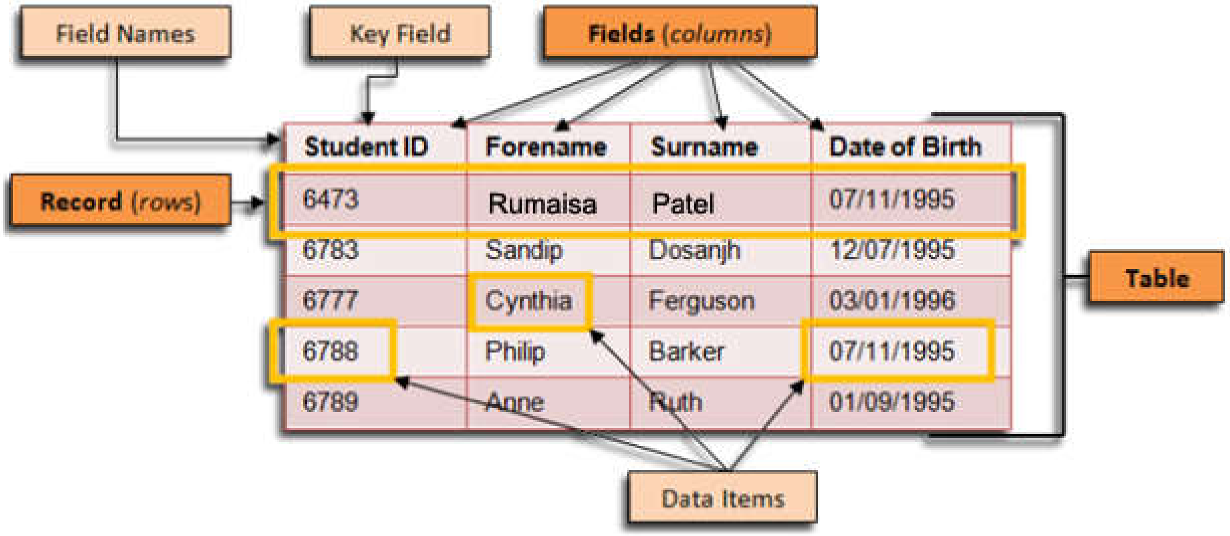
Entity

An entity is a “real world thing” about which data is held. Examples of entities include:

A customer	A product	A pupil	A supplier
A hotel room	A DVD	A flight	A holiday
A treatment	An address book	A book	A car
An order	An animal	A student	

An attribute is a feature of that entity. For example, a hotel room might have an attribute about whether it has a view or whether it is single or double. A student might have a date of birth and an address.

An entity is stored as a table in a database and an attribute becomes a field in a table.
All the data about a particular entity is stored in a single table. Each data item about the entity is a field.



Student ID	Forename	Surname	Date of Birth
6473	Rumaisa	Patel	07/11/1995
6783	Sandip	Dosanjh	12/07/1995
6777	Cynthia	Ferguson	03/01/1996
6788	Philip	Barker	07/11/1995
6789	Anne	Ruth	01/09/1995

Database record

Data in a database table is organised into rows **(records)** and columns **(fields)**. Each record in a relational database table corresponds to an entity. In the example table of 'Students' above there are 5 records. Each record corresponds to an individual student. Note that although there are two students called Philip Barker with the same date of birth, they have different Student IDs and are different students.

Database field

An attribute is a piece of information or a characteristic of an entity. Attributes of entities are represented in database tables by **fields** (columns). A field stores one item of data for a record. In the table above, each student is represented in the relational database by a record and the student attributes are stored in the following fields:

- Student ID
- Forename
- Surname
- Date of Birth

Fields have the following characteristics:

- Each field in a table has a unique name. Note, however, that the same field name can occur in other tables of the same relational database.
- Each field stores a single item of data - For example, a field called Date of Birth would store no more than one date of birth value.
- Each field has a particular data type – for example, text, Boolean, integer, date/time, etc.
- Each field can have its own validation rules - these ensure that data recorded in the field is of the right type and format.

Data types

Different data types are identified so that a computer can store and process the data appropriately.

Data types include:

- text (or string)
- number (numeric) may include:
  - Auto number
  - Currency
- date/time
- Boolean (or Yes/No).

Primary Keys

Each table has a primary key. This is a field chosen so that it can uniquely identify each record.

Sometimes an existing attribute can be used because it is unique but most of the time some sort of ID is created. Primary keys can be used to link to foreign keys in other tables. A foreign key is the primary key in a different table and it is not necessarily unique.

Example Question:

A picture gallery owner has decided to set up a database to keep information about the pictures he has for sale. The database table, PICTURE, will contain the following fields: Title; Artist; Description; Catalogue Number; Size (area in square centimeters); Price; Arrived (date picture arrived at gallery); Sold (whether picture is already sold)

(a) (i) State what data type you would choose for each field.

Title: ....

Artist: ....

Description: ....

Catalogue Number: ....

Size: ....

Price: ....

Arrived: ....

Sold: ....[4]

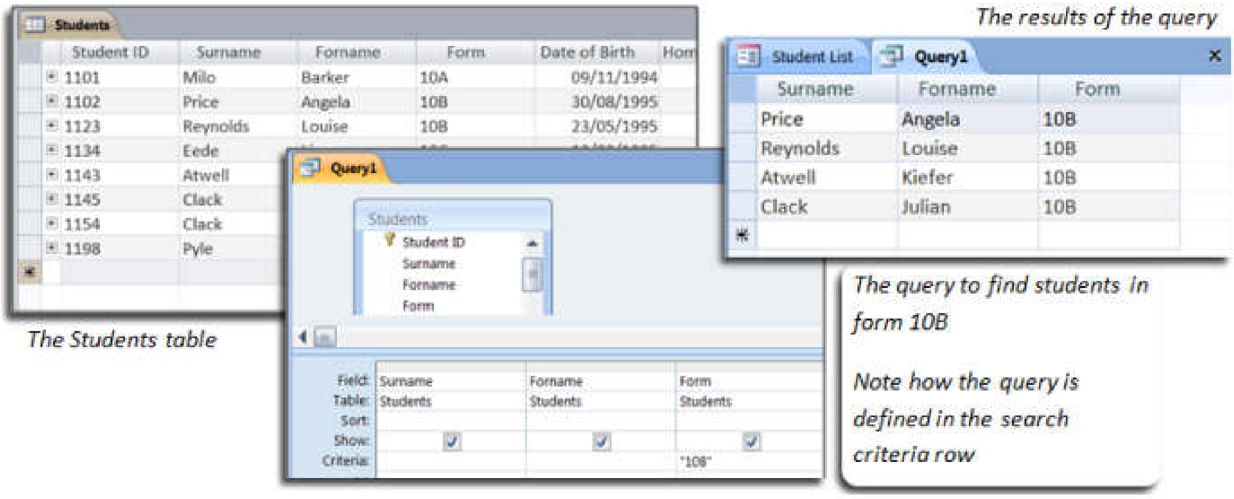
(ii) State which field you would choose for the primary key

…………………………………………………………………………………....................[1]

Query
The prime function of a relational database is to store data in an organised way so that users can interrogate (search) and manipulate (sort) the data. The interrogation of a database is called querying the database and a question used to interrogate the data is called a query. **Query by Example** (**QBE**) is a database **query** language for relational databases. It was devised by Moshé M. Zloof at IBM Research during the mid-1970s, in parallel to the development of SQL. It is the first graphical **query** language, using visual tables where the user would enter commands, **example** elements and conditions.

Databaseuser-interface in which the user fills out a form to retrieve data. The database makes the search on the basis of the example(s) provided by the user.

The query to find students in form 10B



Students

Student ID	Surname	Forname	Form	Date of Birth	Hom
1101	Milo	Barker	10A	09/11/1994	
1102	Price	Angela	10B	30/08/1995	
1123	Reynolds	Louise	10B	23/05/1995	
1134	Eede				
1143	Atwell				
1145	Clack				
1154	Clack				
1198	Pyle				

The Students table

Query1

Students
- Student ID
- Surname
- Forname
- Form

Field:	Surname	Forname	Form
Table:	Students	Students	Students
Sort:			
Show:	☑	☑	☑
Criteria:			"10B"

The results of the query

Student List · Query1

Surname	Forname	Form
Price	Angela	10B
Reynolds	Louise	10B
Atwell	Kiefer	10B
Clack	Julian	10B

The query to find students in form 10B

Note how the query is defined in the search criteria row

A complex query looks for data in two or more fields and uses the logical operators OR, AND or NOT.

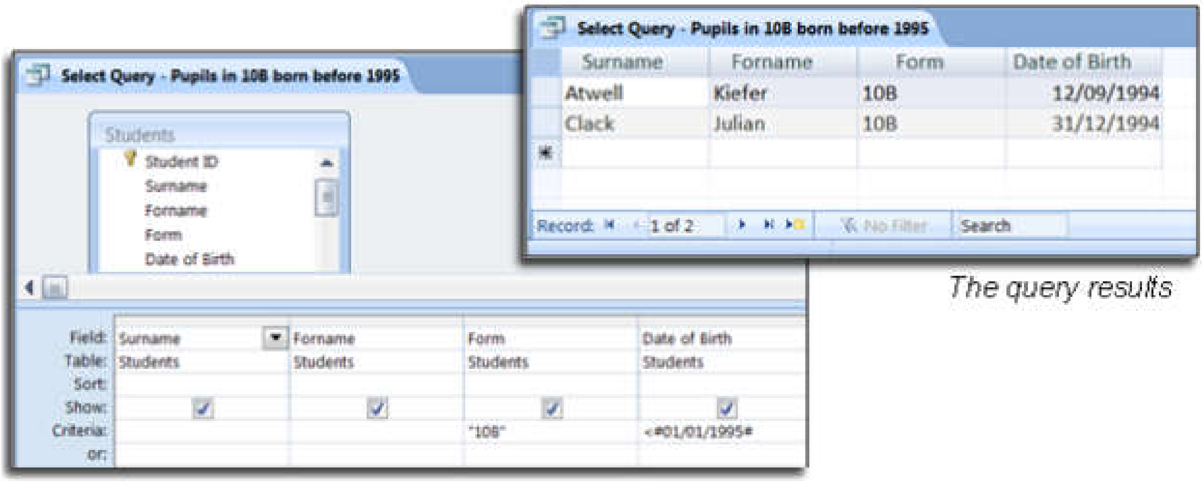
The following example uses a complex query to find all of the pupils in Form 10B who were born before 1995. This query uses the logical operator AND:

(Form = "10B") AND (Date of Birth < 01/01/1995).

Operators can be used to refine search results.

Operator	Meaning
=	Equals
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
<>	Not equal to

The query design is shown below. Note that this time there are two entries in the search criteria row Also note that this time the query has been given a meaningful name ***("Select Query – Pupils in 10B born before 1995").*** This saves other database users from unnecessarily creating the same query.



Select Query - Pupils in 10B born before 1995

Students
- Student ID
- Surname
- Forname
- Form
- Date of Birth

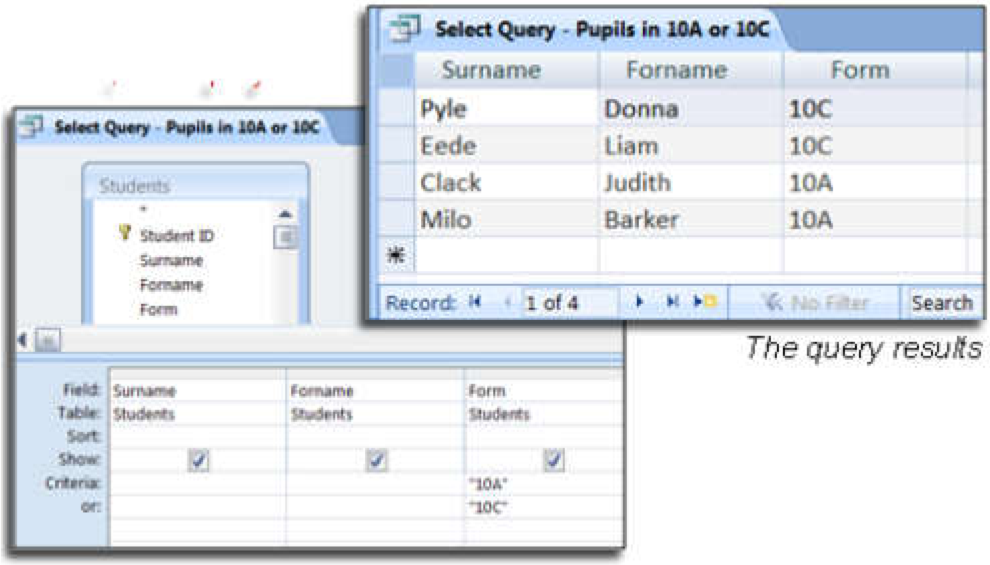
Field:	Surname	Forname	Form	Date of Birth
Table:	Students	Students	Students	Students
Sort:				
Show:	☑	☑	☑	☑
Criteria:			"10B"	<#01/01/1995#
or:				

Select Query - Pupils in 10B born before 1995

Surname	Forname	Form	Date of Birth
Atwell	Kiefer	10B	12/09/1994
Clack	Julian	10B	31/12/1994

The query results

Below is a new complex query that uses the logical operator OR to find pupils who are in Form 10A or Form 10C: (Form = "10A" OR "Form = "10C") this time, in the query definition there will be two criteria lines. The query and its results are shown below:



Select Query - Pupils in 10A or 10C

Students
- *
- Student ID
- Surname
- Forname
- Form

Field:	Surname	Forname	Form
Table:	Students	Students	Students
Sort:			
Show:	☑	☑	☑
Criteria:			"10A"
or:			"10C"

Select Query - Pupils in 10A or 10C

Surname	Forname	Form
Pyle	Donna	10C
Eede	Liam	10C
Clack	Judith	10A
Milo	Barker	10A

The query results

Wildcards in Queries

Wildcard characters can be used in database queries. For example you may want a list of all pupils born in November, or all of the pupils whose surname starts with a 'C'. Wildcard searches allow you to specify the part of the data that you know and leave the data handling software to fill in the blanks.

Surname Like "C*" would find all records where the surname begins with a C**.**

Example Question:

Q 14.1)

The terms **file**, **record** and **field** are used in databases. Explain the meaning of each term and explain the connections between them; you may wish to include a diagram.

....

....

....

Q 14.2) A database was set up to compare oil companies. A section of the database is shown below:

Code	Name of company	No of employees	No of countries	Head office	Profits (billion $)	Share price ($)
AR	Arrows	60 000	30	Americas	8.0	39.00
GZ	Gazjeti	35 000	4	Asia	5.0	44.50
KO	Konoco	40 000	22	Americas	10.0	18.55
OS	Oilbras	56 000	11	Americas	4.0	59.60
SD	Sand Oil	102 000	51	Europe	12.0	15.30
SN	Southern Oil	50 000	15	Americas	11.0	10.90
ST	Static Oil	80 000	31	Americas	10.0	52.05
SU	Summation	70 000	40	Europe	9.0	30.40
WP	Wasp Petrol	90 000	44	Europe	15.0	92.80

(a) Complete the query-by-example grid below to show no. of employees, head office and profit, have a No of countries greater than 40 and Head office is Europe?

Field:					
Table:					
Sort:					
Show:	□	□	□	□	□
Criteria:					
or:					

[5]

(b) Output of the above query will be: ....

.... [2]

(a) Complete the query-by-example grid below to show name of oil companies and code have a share priceless than $50 or whose profits were greater than 8 billion dollars?

Field:					
Table:					
Sort:					
Show:	□	□	□	□	□
Criteria:					
or:					

[5]

(b) Output of the above query will be: ....

.... [2]

Q 14.3) A survey of motorways was carried out and a database was produced. A section of thedatabase is shown below.

Motorway ID	Length (km)	Cars per day	Toll charge per km ($)	Number of lanes
M1	100	50 000	0.60	2
M2	210	75 000	0.40	3
M3	180	60 000	0.50	4
M4	40	20 000	0.30	3
M5	25	15 000	0.10	2
M6	100	40 000	0.70	4
M7	30	10 000	0.40	2
M8	150	60 000	0.60	4

(a) How many fields and how many records are shown?

(i) number of fields ....

(ii) number of records .... [2]

(b) Complete the query-by-example grid below to show Motorway ID and Lanes if the following search condition was used?

(Length (km) > 100) AND (Number of lanes > 3)

Field:					
Table:					
Sort:					
Show:	□	□	□	□	□
Criteria:					
or:					

[5]

(c) Output of the above query will be:..[1]

(d)Complete the query-by-example grid below to show the motorways where the number of cars per day exceeds 50 000 or the toll charge per kilometre is greater than $0.50?

Field:					
Table:					
Sort:					
Show:	□	□	□	□	□
Criteria:					
or:					

(e) Output of the above query will be: ....[2]

Q14.4) An estate agent uses a computer system to store details about properties for sale. A section of the properties file is shown below:

Property	Type	Area	Price
4217	D	Lake	99.92
4219	D	Park	200.00
4220	S	Lake	105.50
4221	F	Park	175.25
4222	F	Town	75.00

a Why is coded data used for Type?

....

....

b What is the data type for Price?

....

....

c What is meant by a key field? Which is the key field?

....

....

Q14.5) Explain the following terms:

database: ....

....

file: ....

....

record ....

....

Q14.6)

Mrs Smith runs a dog sitting service that looks after dogs whose owners are going away on holiday. An extract of the JOB table is shown below:

JobNumber	**DogID**	**JobType**	**Time**	**Details**
35	SM13	Feed	Morning	250 g of Hundex
36	BA12	Walk	Afternoon	At least 30 minutes
37	SM13	Walk	Afternoon	Keep on leash
38	GH14	Other	Morning	Medicine: 1 tablet of Depucine
39	HT19	Other	Evening	Brush fur

Mrs Smith uses a query to select jobs using the following criteria:

(Time = “Afternoon”) OR (Time = “Evening”)

List the JobNumbers of the jobs that will be selected from the extract shown.

....

.... [1]

Q14.7)

A secondary school uses a database to store all requests for IT maintenance.

(a) A database is defined as a persistent store of organised data.

Explain what is meant by ‘a persistent store of organised data’.

....

....

....

.... [2]

(b) The database stores information about the teachers, the hardware devices that each teacher has and the requests that have been made for IT maintenance.

The database has a table called REQUESTS.

An extract of the data in the table REQUESTS is shown in Table:

RequestID	**TeacherID**	**Date**	**Details**	**HardwareID**
0001	VE1	12/04/2017	Laptop battery fault	LAP#121
0002	GC1	12/04/2017	Interactive whiteboard will not connect	INT#002
0003	SO3	13/04/2017	USB drive corrupted	MEM#033
0004	VE1	14/04/2017	Java update needed	LAP#121

(i) Identify the most appropriate data type for the field RequestID, giving a reason for your choice.

Data type ....

Reason ....

.... **[2]**

(ii) State how many records are shown in Table.

.... **[1]**

(iii) Identify the most appropriate field to be the Primary Key, giving a reason for your choice.

Field ....

Reason ....

.... **[2]**

(c) Validation is one feature of a DBMS that can be used to create customised data handlingapplications.

(i) For each of the fields listed below, identify **one** validation rule that could be used. Each rule must be different.

TeacherID ....

....

Date...

.... **[2]**

(ii) Identify and describe **two** additional features of a DBMS that can be used to createcustomised data handling applications, giving an example of how each could be used in

this database.

Feature 1 ....

Description ....

....

Example use ....

....

Feature 2 ....

Description

....

....

Example use ....

.... **[6]**

Q 14.8) A social networking site uses a database to store the details of the people who have joined the site.

(a) Describe what is meant by a database.

....

.... [2]

(b) When a person joins the website, they need to enter some personal data which is validated using rules. For example, the date of birth must be in the past.

State **one** rule that could be used when validating each of the following.

Email address: ....

....

Gender: ....

....

Password: ....

....**[3]**

Examination Questions

14.9 Specimen paper 2016 P2

7 A database was set up to show the properties of certain chemical elements. Part of the database is shown below.

Name of element	Element symbol	Atomic number	Atomic weight	Melting point (C)	Boiling point (C)	State at room temp
oxygen	O	8	16	–218	–183	gas
iron	Fe	26	56	1538	2861	solid
mercury	Hg	80	201	–38	356	liquid
bromine	Br	35	80	–7	59	liquid
osmium	Os	76	190	3033	5012	solid
caesium	Cs	55	133	28	671	solid
gallium	Ga	31	70	30	2204	solid
argon	Ar	18	40	–189	–186	gas
silver	Ag	47	108	961	2162	solid

(a) How many fields are in each record?

.... [1]

(b) The following search condition was entered:

(Melting point (C) < 40) AND (Atomic weight > 100)

Using Element symbol only, which records would be output?

....

.... [2]

(c) Which field would be best suited as primary key?

....[1]

14.10 Summer 2015 P21& 23

7 A database, PROPERTY, was set up to show the prices of properties for sale and the features of each property. Part of the database is shown below.

Property Type	Brochure No	Number of Bedrooms	Number of Bathrooms	Garden	Garage	Price in $
Bungalow	B17	7	4	Yes	Yes	750,000
Apartment	A09	2	1	No	No	100,000
House	H10	4	2	Yes	No	450,000
House	H13	3	2	Yes	No	399000
Apartment	A01	2	2	No	Yes	95000
Apartment	A16	1	1	No	No	150000
House	H23	3	1	No	Yes	250000
House	H46	2	1	Yes	Yes	175000

(a) Give the number of fields that are in each record.

....[1]

(b) State which field you would choose for the primary key.

....

Give a reason for choosing this field.

....

....[2]

(c) State the data type you would choose for each of the following fields.

Garage ....

Number of Bedrooms ....

Price in $ ....[3]

(d) The query-by-example grid below selects all houses with more than 1 bathroom and more than 2 bedrooms.

Field:	Property Type	Number of Bedrooms	Number of Bathrooms	Price in $	Brochure No
Table:	PROPERTY	PROPERTY	PROPERTY	PROPERTY	PROPERTY
Sort:				Ascending	
Show:	□	□	□	☑	☑
Criteria:	= 'House'	>2	>1		
or:					

Show what would be output.

....

....[2]

(e) Complete the query-by-example grid below to select and show the brochure number, property type and price of all properties with a garage below $200,000.

Field:				
Table:				
Sort:				
Show:	□	□	□	□
Criteria:				
or:				

[4]

Examiner Report Question 7
(a) Many candidates correctly identified the number of fields in each record.
(b) Most candidates correctly identified the field to choose for the primary key. Better candidates gave a correct reason for their choice.
(c) Nearly all candidates correctly stated at least one data type.
(d) Most candidates correctly showed only the Price in $ and the Brochure No, as identified by the query-by-example grid. Better candidates showed attention to detail, by correctly putting the prices in ascending order and the Price in $ field before the Brochure No field as indicated by the query-by-example grid.
(e) Most candidates correctly identified the fields to include in the query-by-example grid and identified those that were to be shown. A common error was to incorrectly set the criterion for the garage, when the data type had been set as a Boolean field in part (c).

14.11 Summer 2015 P22

6 A database, MARKS, was set up to record the test results for a class of students. Part of the database is shown below.

Student Name	Class ID	Maths	English	Science	History	Geography
Paul Smith	0017	70	55	65	62	59
Ravi Gupta	0009	29	34	38	41	44
Chin Hwee	0010	43	47	50	45	52
John Jones	0013	37	67	21	28	35
Diana Abur	0001	92	88	95	89	78
Rosanna King	0016	21	13	11	27	15

(a) Give the number of fields that are in each record.

....[1]

(b) State which fields you would choose for the primary key.

....

Give a reason for choosing this field.

....

....[2]

(c) The query-by-example grid below selects all students with more than 60 marks in History or more than 60 marks in Geography.

Field:	Student Name	History	Geography
Table:	MARKS	MARKS	MARKS
Sort:	Ascending		
Show:	☑	□	□
Criteria:		>60	
or:			>60

Show what would be output.

....

....[2]

(d) Complete the query-by-example grid below to select and show the student names only of all students with less than 40 marks in both Maths and English. [3]

Field:			
Table:			
Sort:			
Show:	□	□	□
Criteria:			
or:			

<u>Examiner's comments on Question 6</u>

(a) Many candidates correctly identified the number of fields in each record.

(b) Most candidates correctly identified the field to choose for the primary key. Better candidates gave a correct reason for their choice.

(c) Better candidates correctly showed only the student names as identified by the query-by-example grid. Some of these candidates correctly ordered the names in ascending order.

(d) Most candidates correctly identified the fields to include in the query-by-example grid and identified those that were to be shown. A common error was to set the Maths or English criteria to OR rather than AND, where both criteria are on the same row.

14.12 Winter 2015 P21 & 22

6 A picture gallery owner has decided to set up a database to keep information about the pictures he has for sale. The database table, PICTURE, will contain the following fields:
Title; Artist; Description; Catalogue Number; Size (area in square centimetres); Price; Arrived (date picture arrived at gallery); Sold (whether picture is already sold)

(a) (i) State what data type you would choose for each field.

Title ....
Artist ....
Description ....
Catalogue Number ....
Size ....
Price ....
Arrived ....
Sold ....[4]

(ii) State which field you would choose for the primary key.

....[1]

(b) Give a validation check that you can perform on each of these fields. Each validation check must be different.

Catalogue Number ....
Size ....
Price ....
Arrived ....[4]

(c) Complete the query-by-example grid below to select and show the Catalogue Number, Title and Price of all unsold pictures by the artist 'Twister'.

Field:					
Table:					
Sort:					
Show:	□	□	□	□	□
Criteria:					
or:					

[5]

Examiners' Comments Question 6

(a) (i) Most candidates correctly identified the correct data type for some of the fields. Candidates who did less well throughout, incorrectly used data types from programming rather than database management.
(ii) Most candidates correctly identified the field to choose for the primary key.
(b) Many candidates correctly identified at least one suitable validation check. Candidates with stronger responses throughout identified four different checks; a few candidates incorrectly repeated a validation check.
(c) Many candidates correctly identified the fields to include in the query-by-example grid; stronger responses identified those fields that were to be shown. A common error was to not include the table name.

14.13 Winter 2015 P23

5 A motor boat hire company decides to set up a database to keep information about boats that are available for hire. The database table, BOAT, will contain the following fields:

Boat Name; Model; Engine Power (in hp); Number of Seats; Life Raft (whether there is a life raft kept on the boat); Day Price (price for a day's hire).

(a) Give the data type you would choose for each field.

Boat Name ....
Model ....
Engine Power ....
Number of Seats ....
Life Raft ....
Day Price ....[3]

(b) State a validation check that you can perform on each of these fields. Each validation check must be different.

Boat Name ....
Model ....
Number of Seats ....
Day Price ....[4]

(c) Complete the query-by-example grid below to select and show the Boat Name, Model and Day Price of a day's hire for all boats with 4 seats and an Engine Power of more than 100 hp.

Field:					
Table:					
Sort:					
Show:	□	□	□	□	□
Criteria:					
or:					

[5]

14.14 Summer 2016 P21 &P23

6 A database, STAFFPHONE, was set up to show the telephone extension numbers for membersof staff working in a department store.

Name	**Department**	**Extension number**
Jane Smith	Toys	129
Sue Wong	Books	124
David Chow	Toys	129
Amy Tang	Household	123
Joe Higgs	Books	124
Jane Smith	Shoes	125
Adel Abur	Shoes	125
Peter Patel	Toys	129

(a) Explain why none of the fields in the database can be used as a primary key.

....

....

....[2]

(b) State a field that could be added as a primary key.

....

Give a reason for choosing this field.

....

.... [2]

(c) Use the query-by-example grid below to provide a list of all members of staff, in alphabetical order, grouped by department. [5]

Field:				
Table:				
Sort:				
Show:	□	□	□	□
Criteria:				
or:				

14.15 Summer 2016 P22

7 A database, SOFASELECT, was set up to show the prices of suites, sofas and chairs for sale from an online furniture warehouse. Part of the database is shown below.

Description	Brochure Number	Number of Seats	Number of Pieces	Material	Colour	Price in $
Sofa	SF17	2	1	Leather	Red	950
Sofa	SF19	3	1	Vinyl	Black	1,000
Suite	SU10	4	3	Velvet	Green	1,500
Suite	SU23	5	3	Leather	Brown	950
Recliner chair	RC01	1	1	Leather	Cream	600
Chair	CH16	1	1	Vinyl	Red	250
Recliner sofa	RS23	4	1	Leather	Cream	1,200
Chair	CH10	1	1	Velvet	Red	175

(a) How many fields are in each record?

....[1]

(b) State which field you would choose for the primary key.

....

Give a reason for choosing this field.

....

....[2]

(c) State the data type you would choose for each of the following fields.

Number of Seats ....

Price in $ ....[2]

(d) The query-by-example grid below selects all the furniture in cream leather.

Field:	Description	Material	Colour	Price in $	Brochure Number
Table:	SOFASELECT	SOFASELECT	SOFASELECT	SOFASELECT	SOFASELECT
Sort:				Descending	
Show:	☑	□	□	☑	☑
Criteria:		= 'Leather'	= 'Cream'		
or:					

Show the output from the query-by-example.

....

....[3]

(e) Complete the query-by-example grid below to select and show the brochure number, material, colour and price of all the furniture with 3 or more seats.

Field:					
Table:					
Sort:					
Show:	□	□	□	□	□
Criteria:					
or:					

[5]

14.16 Winter 2016 P21-23

6 A database, THEATRETOURS, was set up to show the tour dates, towns, number of seats and prices in local currency for a Shakespeare play.

Town	**Tour Date**	**Number of Seats**	**Price Local Currency**
Wigan	18/08/2016	120	15.00
Dumfries	20/08/2016	160	12.50
Turin	25/08/2016	200	17.00
Macon	27/08/2016	75	18.00
Bordeaux	29/08/2016	170	20.00
Algiers	01/09/2016	125	1350.00
Windhoek	05/09/2016	65	90.00
Windhoek	06/09/2016	65	90.00
Port Elizabeth	10/09/2016	200	110.00

(a) Explain why none of the fields in the database can be used as a primary key.

....

....

....[2]

(b) State a field that could be added as a primary key.

....

Give a reason for choosing this field.

....

....[2]

(c) Use the query-by-example grid below to provide a list of tour dates and seat prices inalphabetical order of town. [4]

Field					
Table					
Sort					
Show:					
Criteria					
or					

14.17 Winter 2016 P22

5 A database, PLAYPRODUCTION, was set up to show the performance dates, prices and number of seats available at a theatre specialising in Shakespeare productions.

Play	**Performance Date**	**Number Seats Stalls**	**Number Seats Circle**	**Price Stalls Seats $**	**Price Circle Seats $**
As You Like It	01/07/2016	120	90	20.00	30.00
As You Like It	02/07/2016	85	45	30.00	40.00
As You Like It	09/07/2016	31	4	30.00	40.00
Macbeth	14/07/2016	101	56	25.00	35.00
Macbeth	15/07/2016	50	34	25.00	35.00
Macbeth	16/07/2016	12	5	35.00	50.00
Julius Caesar	22/07/2016	67	111	20.00	20.00
Julius Caesar	23/07/2016	21	24	15.00	15.00
A Comedy of Errors	30/07/2016	45	36	35.00	45.00

(a) Give the number of fields that are in each record.

.... [1]

(b) State the data type you would choose for each of the following fields.

Play ....

Number Seats Stalls ....

Price Stalls Seats $ .... [3]

(c) The query-by-example grid below selects all the productions with more than 100 seats left in
either the stalls or the circle.

Field:	Play	Performance Date	Number Seats Stalls	Number Seats Circle
Table:	PLAYPRODUCTION	PLAYPRODUCTION	PLAYPRODUCTION	PLAYPRODUCTION
Sort:	Ascending			
Show:	☑	☑	□	□
Criteria:			> 100	
or:				> 100

Show what would be output from the query-by-example.

....

....

.... [3]

(d) Complete the query-by-example grid below to select all the productions with at least six seatsleft in the circle and show the Play, Performance Date and Price Circle Seats $ in Performance
Date order. [5]

Field:					
Table:					
Sort:					
Show:	□	□	□	□	□
Criteria:					
or:					

14.18 March 2017 P21 (India)

6 A database table, DEVICE, has been set up to record the electronic equipment used in a smallbusiness.

Device ID	**Device Type**	**User**	**Purchase Date**	**Purchase Price ($)**	**Portable**
3	Desktop	Alan Swales	14/02/2017	1350.00	N
4	Laptop	Chantel Law	01/02/2016	1460.00	Y
5	Tablet	Abdula Saud	31/12/2016	1000.00	Y
6	Desktop	Abdula Saud	14/03/2017	1000.00	N
7	Laptop	Alan Swales	15/03/2016	1700.00	Y
8	Tablet	TaonaJaji	16/12/2016	470.00	Y

(a) The query-by-example grid below selects certain records.

Field:	User	Portable	Purchase Price ($)
Table:	DEVICE	DEVICE	DEVICE
Sort:	Ascending		
Show:	☑	□	□
Criteria:		Y	>1000
or:			

Show what would be the output from the query-by-example.

....

....

....[2]

(b) Complete the query-by-example grid below to select all Desktop devices that were either purchased before 31/12/2016 or cost under $1000. Only show the Device ID and DeviceType.

Field:					
Table:					
Sort:					
Show:	□	□	□	□	□
Criteria:					
or:					

14.19 Summer 2017 P21

7 A television (TV) store has a database table, TVSTOCK, for its new range of televisions. The table stores the screen size of each TV, whether it will show 3D, whether the screen is curved or flat, if the internet is available on the TV, if it has a built-in hard disk drive and the price. Part of the database table is shown below.

TVID	ScreenSize	3D	CurvedFlat	Internet	HDD	Price
TV80CVINT	80	YES	CV	YES	YES	$7,000.00
TV65CVINT	65	YES	CV	YES	YES	$5,000.00
TV60CVINT	60	YES	CV	YES	YES	$4,500.00
TV60FTINT	60	YES	FT	YES	YES	$4,000.00
TV55CVINT	55	YES	CV	YES	NO	$3,000.00
TV55FTINT	55	YES	FT	YES	NO	$3,500.00
TV55FTNIN	55	YES	FT	NO	NO	$3,000.00
TV50CVINT	50	YES	CV	YES	NO	$2,500.00
TV50FTINT	50	YES	FT	YES	NO	$2,000.00
TV50FTNIN	50	YES	FT	NO	NO	$1,750.00
TV42FTINT	42	YES	FT	YES	NO	$1,500.00
TV37FTINT	37	NO	FT	YES	NO	$1,200.00
TV20FTNIN	20	NO	FT	NO	NO	$800.00
TV15FTNIN	15	NO	FT	NO	NO	$400.00

(a) State the type of the field **TVID** and give a reason for your choice.

....

....

....[1]

(b) Complete the table with the most appropriate data type for each field. [3]

Field name	**Data type**
ScreenSize	
3D	
CurvedFlat	
Internet	
HDD	
Price	

(c) Use the query-by-example grid below to provide a list of all of the curved screen TVs that have a built-in hard disk drive. Make sure the list only displays the TVID, the price and the screen size in ascending order of price. [5]

Field:					
Table:					
Sort:					
Show:	□	□	□	□	□
Criteria:					
or:					

14.20Summer 2017 P22

5 A database table, SHEEP, is used to keep a record of the sheep on a farm. Each sheep has aunique ear tag, EARnnnn; n is a single digit. The farmer keeps a record of the date of birth, thegender and the current weight of each sheep in kilograms.

(a) Identify the **four** fields required for the database. Give each field a suitable name and datatype. Provide a sample of data that you could expect to see in the field. [8]

Field 1 name..

Data type ....

Data sample ....

Field 2 name..

Data type ....

Data sample ....

Field 3 name..

Data type ....

Data sample ....

Field 4 name..

Data type ....

Data sample ....

(b) State the field that you would choose as the primary key.

....[1]

(c) Using the query-by-example grid below, write a query to identify the ear tags of all malesheep weighing over 10 kilograms. Only display the ear tags.

[3]

Field:				
Table:				
Sort:				
Show:	□	□	□	□
Criteria:				
or:				

14.21 Winter 2017 P21

6 A wildlife park has a database table, called LIVESTOCK, to classify and record its animal species.Part of the database table is shown.

Species	Classification	Diet	Legs
Giraffe	Mammal	Herbivore	4
Elephant	Mammal	Herbivore	4
Crocodile	Reptile	Carnivore	4
Ostrich	Bird	Omnivore	2
Gorilla	Mammal	Herbivore	2
Bear	Mammal	Omnivore	4
Rhinoceros	Mammal	Herbivore	4
Hippopotamus	Mammal	Herbivore	4
Flamingo	Bird	Omnivore	2
Lion	Mammal	Carnivore	4
Turtle	Reptile	Omnivore	4
Penguin	Bird	Carnivore	2

(a) Suggest another appropriate field that could be added to this database by stating its nameand data type. State its purpose and give an example of the data it could contain.

Field name ....

Data Type ....

Purpose ....

....

Example of data ....[2]

(b) Use the query-by-example grid below to provide a list of all four legged mammals that are herbivores, sorted alphabetically by species, with only the species displayed.

Field:					
Table:					
Sort:					
Show:	□	□	□	□	□
Criteria:					
or:					

14.22 Winter 2017 P22

6 A database table, TRAIN, is to be set up for a railway company to keep a record of the engines available for use. Each engine has a unique number made up of 5 digits, nnnnn. The engines are classified as freight (F) or passenger (P) together with a power classification that is a whole number between 0 and 9, for example F8. The railway company keeps a record of the date of the last service for each engine.

(a) Identify the **three** fields required for the database. Give each field a suitable name and data type. Provide a sample of data that you could expect to see in the field.

Field 1 Name ....

Data type ....

Data sample ....

Field 2 Name ....

Data type ....

Data sample ....

Field 3 Name ....

Data type ....

Data sample ....[6]

(b) State the field that you should choose as the primary key.

....[1]

(c) Using the query-by-example grid below, write a query to identify all passenger engines that have not been serviced in the past 12 months. Only display the engine numbers.

[3]

Field:				
Table:				
Sort:				
Show:	□	□	□	□
Criteria:				
or:				

14.23 March 2018 P22 (India)

6 A database table, JEWEL, is used to keep a record of jewellery for sale in a shop. Each item of jewellery can be made of silver, platinum or gold metal. The shop stocks rings, bracelets and necklaces. The number in stock and the price is also stored.

(a) Identify the **four** fields required for the database. Give each field a suitable name and data type. Explain why you chose the data type for each field.

Field 1 Name .... Data type ....

Explanation ....

....

Field 2 Name .... Data type ....

Explanation ....

....

Field 3 Name .... Data type ....

Explanation ....

....

Field 4 Name .... Data type ....

Explanation ....

....[8]

(b) Explain why none of these fields could be used as a primary key.

....

....[1]

(c) Using the query-by-example grid below, write a query to identify the silver bracelets. Only display the number in stock and the price.

Field:				
Table:				
Sort:				
Show:	□	□	□	□
Criteria:				
or:				

Comments on Question 6

(a) *Nearly all candidates identified some appropriate fields and could also provide a suitable data typeand explanation. Many candidates provided excellent answers worth full marks.*

(b) *Nearly all candidates gave a correct explanation as to why none of the fields were suitable use as a primary key.*

(c) *Nearly all candidates correctly identified the fields required in the query-by-example grid. Most candidates correctly identified which fields to show. Many candidates provided suitable criteria to identify that only details of silver bracelets were required.*

14.24Summer 2018 P21

6 A database table, PERFORMANCE, is used to keep a record of the performances at a localtheatre.

Show Number	**Type**	**Title**	**Date**	**Sold Out**
SN091	Comedy	An Evening at Home	01 Sept	Yes
SN102	Drama	Old Places	02 Oct	No
SN113	Jazz	Acoustic Evening	03 Nov	No
SN124	Classical	Mozart Evening	04 Dec	Yes
SN021	Classical	Bach Favourites	01 Feb	Yes
SN032	Jazz	30 Years of Jazz	02 Mar	Yes
SN043	Comedy	Street Night	03 Apr	No
SN054	Comedy	Hoot	04 May	No

(a) State the number of fields and records in the table.

Fields ....

Records ....[2]

(b) Give **two** validation checks that could be performed on the **Show Number** field.

Validation check 1 ....

....

Validation check 2 ....

.... [2]

(c) Using the query-by-example grid, write a query to identify jazz performances that are not soldout. Only display the date and the title. [4]

Field:					
Table:					
Sort:					
Show:	□	□	□	□	□
Criteria:					
or:					

14.25Summer 2018 P21

6 A database table, TREES, is used to keep a record of the trees in a park. Each tree is given aunique number and is examined to see if it is at risk of dying. There are over 900 trees; part of thedatabase table is shown.

Tree Number	Type	Map Position	Age in Years	At Risk
TN091	Acacia	A7	250	Y
TN172	Olive	C5	110	N
TN913	Cedar	B9	8	N
TN824	Banyan	A3	50	Y
TN021	Pine	D5	560	Y
TN532	Teak	C8	76	Y
TN043	Yew	B1	340	N
TN354	Spruce	D4	65	N
TN731	Elm	B10	22	Y
TN869	Oak	C9	13	N
TN954	Pine	E11	3	N

(a) State the number of fields in the table.

....[1]

(b) The tree numbering system uses TN followed by three digits. The numbering system will notwork if there are over 1000 trees.

Describe, with the aid of an example, how you could change the tree numbering system toallow for over 1000 trees. Existing tree numbers must not be changed.

....

....

....

....[2]

(c) Using the query-by-example grid, write a query to identify at risk trees over 100 years old. Display only the type and the position on the map. [4]

Field:					
Table:					
Sort:					
Show:	□	□	□	□	□
Criteria:					
or:					

Q 14.1)

*A **field** is a single piece of information; a **record** is one complete set of fields; and a **file** is a collection of records. For example, a telephone book is analogous to a file. It contains a list of records, each of which consists of three fields: name, address, and telephone number.*

Q 14.6)

• 36, 37, 39 (correct answer only)

Q 14.7)

a)1 mark per bullet to max 2
•□The data is stored permanently / is unchanging / remains when the DBMS or application is closed / non volatile / on secondary storage
•□The data has a structure / stored in tables / fields / records
b i) 1 mark for data type, 1 for justification
Data type: Text/String
Reason: leading 0s/will not be treated as a number
b ii) 4
b iii) 1 mark for Field name, 1 for reason
Field: RequestID
Reason: it will be unique/each request will have a unique number/it will not be repeated/other fields can be repeated
c i) Max 1 mark per validation rule.
Both rules must be different, e.g. cannot both be presence
TeacherID
e.g.
•□Presence check//must be entered
•□Format check//must be letters then numbers
•□Existence check//must already exist in the database
•□Lookup//must be selected from a list of valid teachers
•□Type/character check//must be string
•□Length check//must be (minimum of) 3 characters long
Date
e.g.
•□Range check//must be within certain dates
•□Presence check//must be entered
•□Format/character check//must be DD/MM/YYYY
•□Type check//must be a valid date
•□Length check//must be 8/10 characters
•□Lookup/existence check//must be a valid date (eg from calendar)
c ii) 3 marks for each feature, 1 for identifying, 1 for description, 1 for example use
e.g
•□Query
•□Use to select specific information // find records that match a criteria // search for records // extract data
•□e.g. Find all requests made on a specific date
•□Form
•□User friendly way to enter data // uses drop down boxes etc.
•□e.g. form to enter a new request
•□Report
•□User friendly/formatted copy of results // can be used as a hard copy // method of outputting data
•□e.g. report of all requests made by one teacher for printing
•□Security
•□Stop unauthorised access or modification
•□e.g. usernames and passwords

14.8 Marking Scheme

(a) *• A persistent...*
• ... and structured/organised store of data
• Allows data to be queried/interrogated.
*b) egEmail address: **Character Check***
• Must contain an @ sign
• Must contain a full stop (after the @ sign).
*Gender: **Look-up Check***
• Must be one of Male, Female, (Other).
*Password: **Length check, type check***
•Must have a given minimum length
•Must contain a non-letter.

14.9 Specimen paper 2016 P2

7 **(a)** 7

(b) Hg, Cs

(c) Element symbol

14.10 Summer 2015 P21& 23

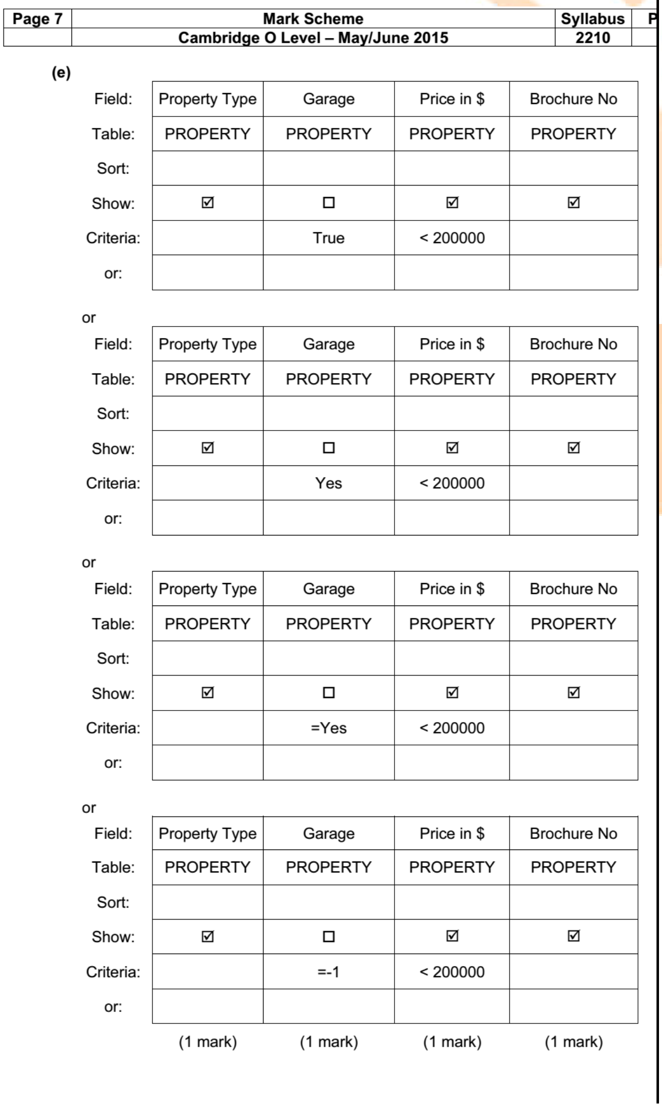
in part (c).

7 **(a)** – 7

(b) – Brochure No
– Uniquely identifies each property

(c) Garage – Boolean
Number of Bedrooms – Number/Integer/Single
Price in $ – Number/Single/Real/Currency

(d) 399000 H13
450000 H10



Page 7	Mark Scheme	Syllabus	P
	Cambridge O Level – May/June 2015	2210	

(e)

Field:	Property Type	Garage	Price in $	Brochure No
Table:	PROPERTY	PROPERTY	PROPERTY	PROPERTY
Sort:				
Show:	☑	□	☑	☑
Criteria:		True	< 200000	
or:				

or

Field:	Property Type	Garage	Price in $	Brochure No
Table:	PROPERTY	PROPERTY	PROPERTY	PROPERTY
Sort:				
Show:	☑	□	☑	☑
Criteria:		Yes	< 200000	
or:				

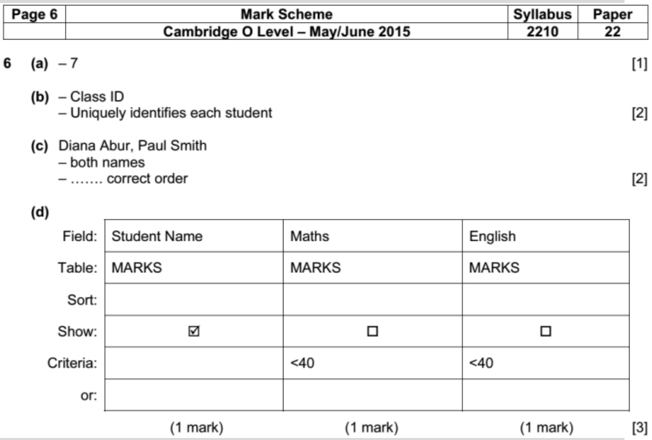
or

Field:	Property Type	Garage	Price in $	Brochure No
Table:	PROPERTY	PROPERTY	PROPERTY	PROPERTY
Sort:				
Show:	☑	□	☑	☑
Criteria:		=Yes	< 200000	
or:				

or

Field:	Property Type	Garage	Price in $	Brochure No
Table:	PROPERTY	PROPERTY	PROPERTY	PROPERTY
Sort:				
Show:	☑	□	☑	☑
Criteria:		=-1	< 200000	
or:				
	(1 mark)	(1 mark)	(1 mark)	(1 mark)

14.11 Summer 2015 P22



Page 6	Mark Scheme	Syllabus	Paper
	Cambridge O Level – May/June 2015	2210	22

6 **(a)** – 7 [1]

(b) – Class ID
– Uniquely identifies each student [2]

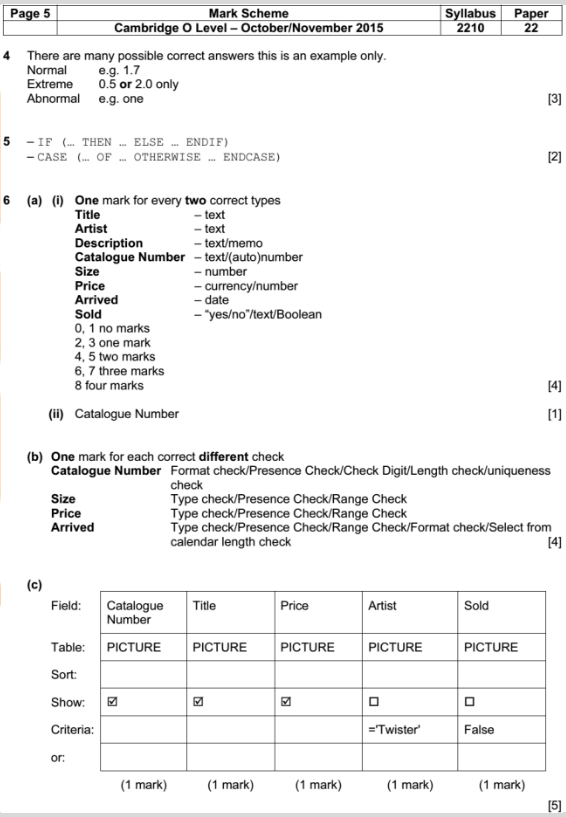
(c) Diana Abur, Paul Smith
– both names
– ……. correct order [2]

(d)

Field:	Student Name	Maths	English
Table:	MARKS	MARKS	MARKS
Sort:			
Show:	☑	□	□
Criteria:		<40	<40
or:			
	(1 mark)	(1 mark)	(1 mark)

[3]

14.12 Winter 2015 P21 & 22



Page 5	Mark Scheme	Syllabus	Paper
	Cambridge O Level – October/November 2015	2210	22

4 There are many possible correct answers this is an example only.

Normal	e.g. 1.7
Extreme	0.5 **or** 2.0 only
Abnormal	e.g. one

[3]

5

```
– IF (… THEN … ELSE … ENDIF)
– CASE (… OF … OTHERWISE … ENDCASE)
```

[2]

6 **(a)** **(i)** **One** mark for every **two** correct types

Title	– text
Artist	– text
Description	– text/memo
Catalogue Number	– text/(auto)number
Size	– number
Price	– currency/number
Arrived	– date
Sold	– "yes/no"/text/Boolean

0, 1 no marks
2, 3 one mark
4, 5 two marks
6, 7 three marks
8 four marks [4]

(ii) Catalogue Number [1]

(b) **One** mark for each correct **different** check

Catalogue Number	Format check/Presence Check/Check Digit/Length check/uniqueness check
Size	Type check/Presence Check/Range Check
Price	Type check/Presence Check/Range Check
Arrived	Type check/Presence Check/Range Check/Format check/Select from calendar length check

[4]

(c)

Field:	Catalogue Number	Title	Price	Artist	Sold
Table:	PICTURE	PICTURE	PICTURE	PICTURE	PICTURE
Sort:					
Show:	☑	☑	☑	□	□
Criteria:				='Twister'	False
or:					
	(1 mark)	(1 mark)	(1 mark)	(1 mark)	(1 mark)

[5]

14.13 Winter 2015 P23

Page 5	Mark Scheme Cambridge O Level – October/November 2015	Syllabus 2210	Paper 23

5 **One** mark for every **two** correct types

Boat Name	–	text
Model	–	text
Engine Power	–	number
Number of Seats	–	number
Life Raft	–	"yes/no"/text/Boolean
Day Price	–	currency/number

0, 1 no marks
2, 3 one mark
4, 5 two marks
6 three marks [3]

(b) **One** mark for each correct **different** check

Boat Name	Presence Check/Type Check/Character Check
Model	Format check/Type check/Presence Check/Length check/ Use of Drop-down box to select
Number of Seats	Type check/Presence Check/Range Check/ Use of Drop-down box to select
Day Price	Type check/Presence Check/Range Check

[4]

(c)

Field:	Boat Name	Model	Day Price	Number of Seats	Engine Power
Table:	BOAT	BOAT	BOAT	BOAT	BOAT
Sort:					
Show:	☑	☑	☑	☐	☐
Criteria:				= 4	> 100
or:					
	(1 mark)	(1mark)	(1 mark)	(1 mark)	(1 mark)

[5]

14.14 Summer 2016 P21 &P23

6 **(a)** – all (fields) have (1 mark) duplicate entries (1 mark)

– none (of the fields) (1 mark) have unique entries(1 mark)

[2]

(b) – e.g. StaffNumber ….

– ….. Uniquely identifies each member of staff//no duplicates//different for each member of staff

[2]

Page 6	Mark Scheme Cambridge IGCSE – May/June 2016	Syllabus 0478	Paper 21

(c)

Field:	Department	Name		
Table:	STAFFPHONE	STAFFPHONE		
Sort:	Ascending	Ascending		
Show:	☑	☑	☐	☐
Criteria:				
or:				
	(2 marks)	(2 marks)	(1 mark for correct order and number of fields shown)	

[5]

14.15 Summer 2016 P22

7 **(a)** – 7

(b) – Brochure Number……..
– ….. Uniquely identifies each record/each Brochure Number different/no duplicates

(c) – Number of Seats – number/integer
– Price in $ – currency/real

(d) **1** mark for each correct result, **1** mark for the results in descending order of price

– Recliner sofa	1,200	RS23
– Recliner chair	600	RC01

Page 6	Mark Scheme Cambridge IGCSE – May/June 2016	Syllabus 0478	Paper 22

(e)

Field:	Brochure Number	Material	Colour	Price in $	Number of Seats
Table:	SOFASELECT	SOFASELECT	SOFASELECT	SOFASELECT	SOFASELECT
Sort:					
Show:	☑	☑	☑	☑	☐
Criteria:					>2
or:					
	(1 mark)	(1 mark)	(1 mark)	(1 mark)	(1 mark)

14.16 Winter 2016 P21-23

6 **(a)** – Town has duplicate entries/all fields can have duplicate entries
– fields other than Town not suitable identifiers

(b) – Performance number …
– … uniquely identifies each performance

Page 5	Mark Scheme Cambridge IGCSE – October/November 2016	Syllabus 0478	Paper 23

(c)

Field:	Town	Tour Date	Price Local Currency	
Table:	THEATRETOURS	THEATRETOURS	THEATRETOURS	
Sort:	Ascending			
Show:	☑	☑	☑	☐
Criteria:				
or:				

1 mark for each column + 1 mark for correct fields only [4]

14.17 Winter 2016 P22

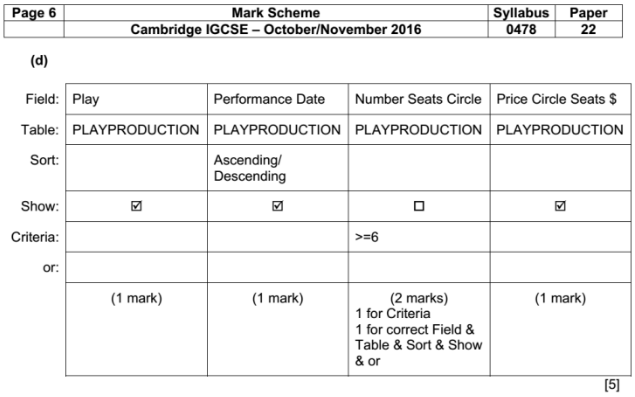
5 **(a)** – 6

(b)

– Play	text
– No Seats Stalls	number
– Price Stalls Seats $	currency

(c) **1 mark** for **correct plays**, **1 mark** for **cor**
text, **1 mark** for the **order**

As You Like It	01/07/2016
Julius Caesar	22/07/2016
Macbeth	14/07/2016



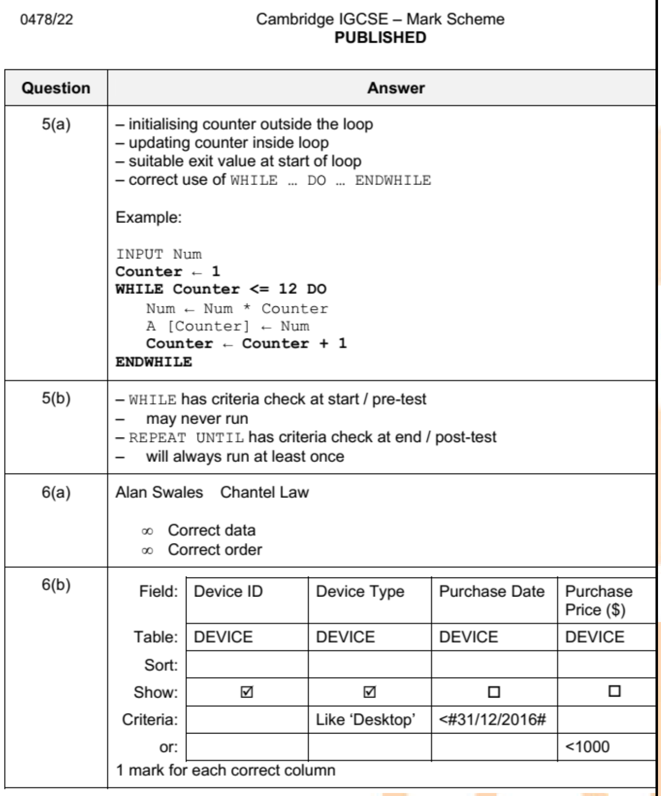
Page 6	Mark Scheme	Syllabus	Paper
	Cambridge IGCSE – October/November 2016	0478	22

(d)

Field:	Play	Performance Date	Number Seats Circle	Price Circle Seats $
Table:	PLAYPRODUCTION	PLAYPRODUCTION	PLAYPRODUCTION	PLAYPRODUCTION
Sort:		Ascending/ Descending		
Show:	☑	☑	□	☑
Criteria:			>=6	
or:				
	(1 mark)	(1 mark)	(2 marks) 1 for Criteria 1 for correct Field & Table & Sort & Show & or	(1 mark)

[5]

14.18 March 2017 P21 (India)



0478/22 Cambridge IGCSE – Mark Scheme
PUBLISHED

Question	Answer

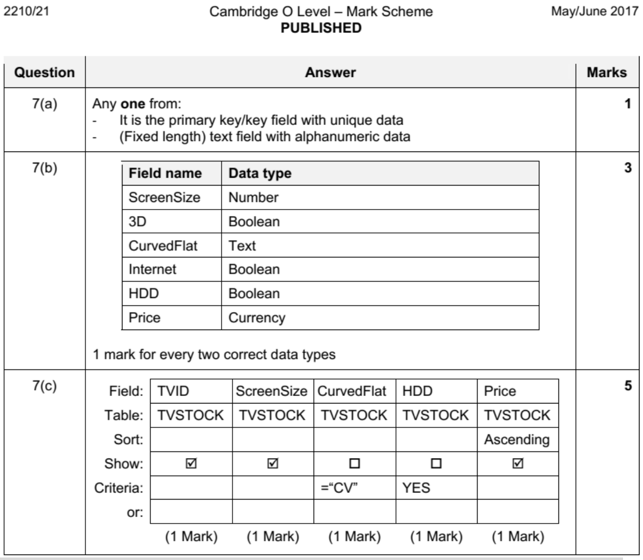
5(a)	– initialising counter outside the loop
	– updating counter inside loop
	– suitable exit value at start of loop
	– correct use of `WHILE … DO … ENDWHILE`
	
	Example:
	
	`INPUT Num`
	`Counter ← 1`
	`WHILE Counter <= 12 DO`
	`    Num ← Num * Counter`
	`    A [Counter] ← Num`
	`    Counter ← Counter + 1`
	`ENDWHILE`

5(b)	– `WHILE` has criteria check at start / pre-test – may never run – `REPEAT UNTIL` has criteria check at end / post-test – will always run at least once
6(a)	Alan Swales Chantel Law ∞ Correct data ∞ Correct order

6(b)	Field:	Device ID	Device Type	Purchase Date	Purchase Price ($)
	Table:	DEVICE	DEVICE	DEVICE	DEVICE
	Sort:				
	Show:	☑	☑	□	□
	Criteria:		Like 'Desktop'	<#31/12/2016#	
	or:				<1000

1 mark for each correct column

14.19 Summer 2017 P21



2210/21 Cambridge O Level – Mark Scheme May/June 2017
PUBLISHED

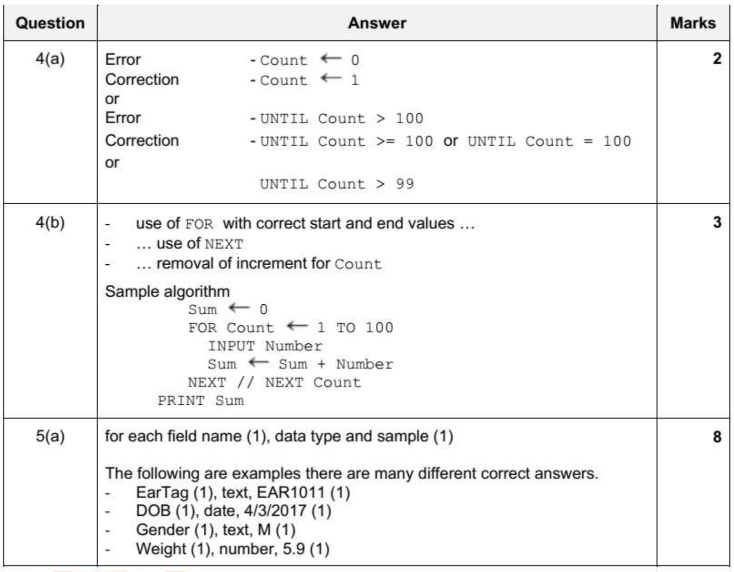
Question	Answer	Marks
7(a)	Any **one** from: - It is the primary key/key field with unique data - (Fixed length) text field with alphanumeric data	1

Question	Field name	Data type	Marks
7(b)	ScreenSize	Number	3
	3D	Boolean	
	CurvedFlat	Text	
	Internet	Boolean	
	HDD	Boolean	
	Price	Currency	

1 mark for every two correct data types

7(c)	Field:	TVID	ScreenSize	CurvedFlat	HDD	Price	5
	Table:	TVSTOCK	TVSTOCK	TVSTOCK	TVSTOCK	TVSTOCK	
	Sort:					Ascending	
	Show:	☑	☑	□	□	☑	
	Criteria:			="CV"	YES		
	or:						
		(1 Mark)	(1 Mark)	(1 Mark)	(1 Mark)	(1 Mark)	

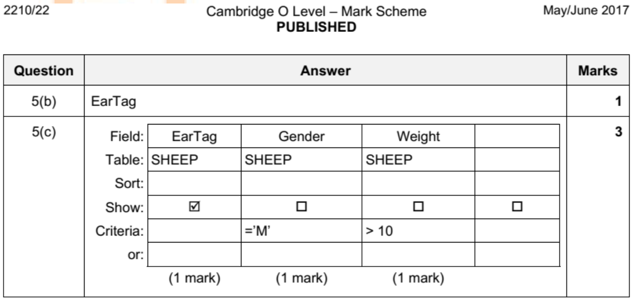
14.20Summer 2017 P22



Question	Answer		Marks
4(a)	Error Correction or Error Correction or	- `Count ← 0` - `Count ← 1` - `UNTIL Count > 100` - `UNTIL Count >= 100` or `UNTIL Count = 100` `UNTIL Count > 99`	2

4(b)	- use of `FOR` with correct start and end values …
	- … use of `NEXT`
	- … removal of increment for `Count`
	
	Sample algorithm
	`    Sum ← 0`
	`    FOR Count ← 1 TO 100`
	`      INPUT Number`
	`      Sum ← Sum + Number`
	`    NEXT // NEXT Count`
	`  PRINT Sum`
			3

5(a)	for each field name (1), data type and sample (1) The following are examples there are many different correct answers. - EarTag (1), text, EAR1011 (1) - DOB (1), date, 4/3/2017 (1) - Gender (1), text, M (1) - Weight (1), number, 5.9 (1)		8

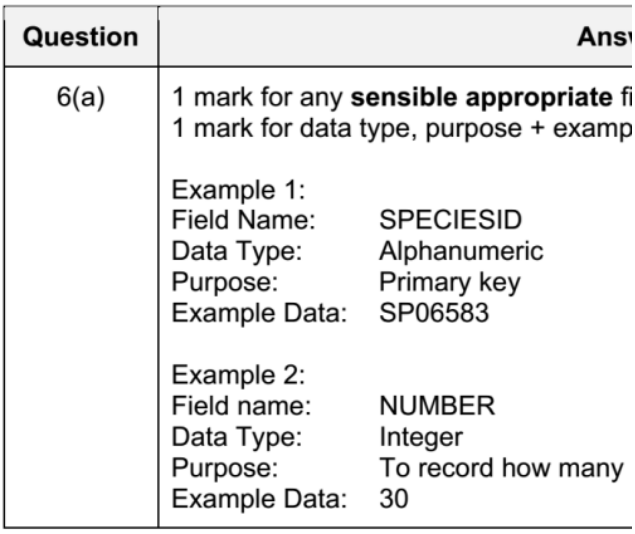


2210/22 Cambridge O Level – Mark Scheme May/June 2017
PUBLISHED

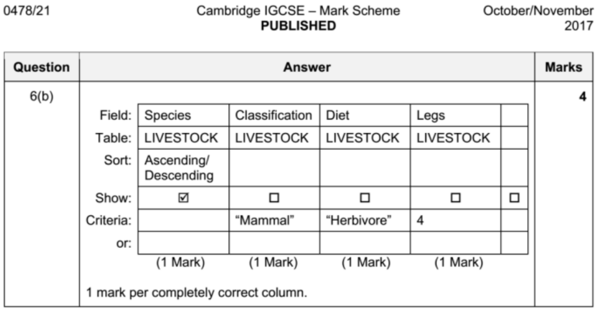
Question	Answer	Marks
5(b)	EarTag	1

5(c)	Field:	EarTag	Gender	Weight		3
	Table:	SHEEP	SHEEP	SHEEP		
	Sort:					
	Show:	☑	□	□	□	
	Criteria:		='M'	> 10		
	or:					
		(1 mark)	(1 mark)	(1 mark)		

14.21 Winter 2017 P21



Question	Answ
6(a)	1 mark for any **sensible appropriate** fi 1 mark for data type, purpose + examp Example 1: Field Name: SPECIESID Data Type: Alphanumeric Purpose: Primary key Example Data: SP06583 Example 2: Field name: NUMBER Data Type: Integer Purpose: To record how many Example Data: 30



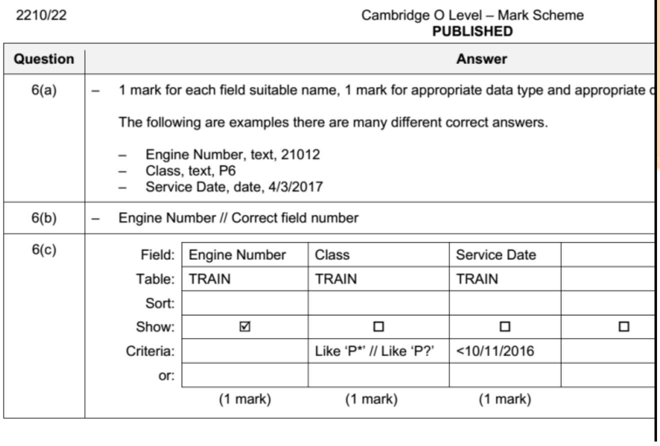
0478/21 Cambridge IGCSE – Mark Scheme **PUBLISHED** October/November 2017

Question	Answer	Marks
6(b)	(see grid below) 1 mark per completely correct column.	4

Field:	Species	Classification	Diet	Legs	
Table:	LIVESTOCK	LIVESTOCK	LIVESTOCK	LIVESTOCK	
Sort:	Ascending/ Descending				
Show:	☑	□	□	□	□
Criteria:		"Mammal"	"Herbivore"	4	
or:					
	(1 Mark)	(1 Mark)	(1 Mark)	(1 Mark)	

1 mark per completely correct column.

14.22 Winter 2017 P22

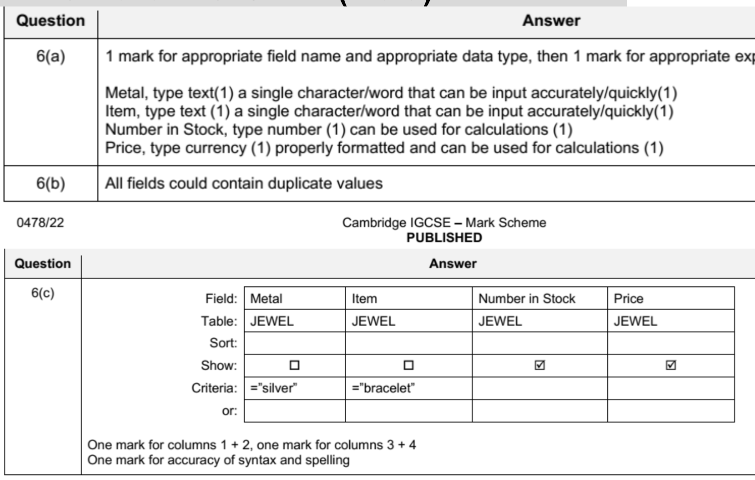


2210/22 Cambridge O Level – Mark Scheme **PUBLISHED**

Question	Answer
6(a)	– 1 mark for each field suitable name, 1 mark for appropriate data type and appropriate c The following are examples there are many different correct answers. – Engine Number, text, 21012 – Class, text, P6 – Service Date, date, 4/3/2017
6(b)	– Engine Number // Correct field number
6(c)	(see grid below)

Field:	Engine Number	Class	Service Date	
Table:	TRAIN	TRAIN	TRAIN	
Sort:				
Show:	☑	□	□	□
Criteria:		Like 'P*' // Like 'P?'	<10/11/2016	
or:				
	(1 mark)	(1 mark)	(1 mark)	

14.23 March 2018 P22 (India)



Question	Answer
6(a)	1 mark for appropriate field name and appropriate data type, then 1 mark for appropriate exp Metal, type text(1) a single character/word that can be input accurately/quickly(1) Item, type text (1) a single character/word that can be input accurately/quickly(1) Number in Stock, type number (1) can be used for calculations (1) Price, type currency (1) properly formatted and can be used for calculations (1)
6(b)	All fields could contain duplicate values

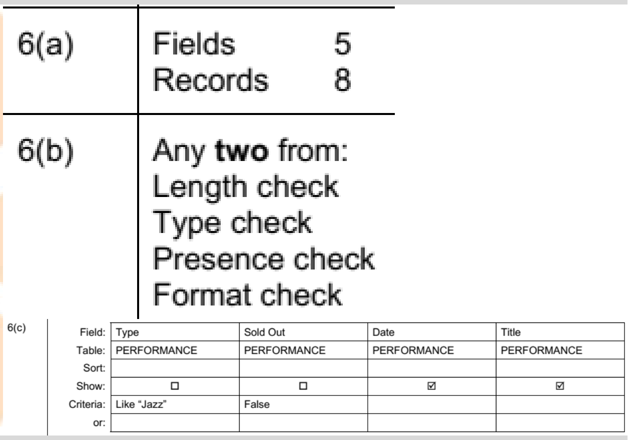
0478/22 Cambridge IGCSE – Mark Scheme **PUBLISHED**

Question	Answer
6(c)	(see grid below) One mark for columns 1 + 2, one mark for columns 3 + 4 One mark for accuracy of syntax and spelling

Field:	Metal	Item	Number in Stock	Price
Table:	JEWEL	JEWEL	JEWEL	JEWEL
Sort:				
Show:	□	□	☑	☑
Criteria:	="silver"	="bracelet"		
or:				

One mark for columns 1 + 2, one mark for columns 3 + 4
One mark for accuracy of syntax and spelling

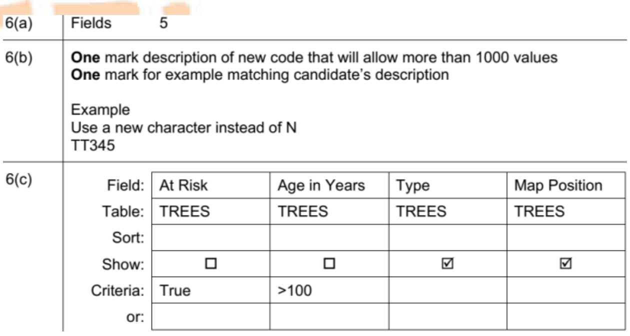
14.24Summer 2018 P21



6(a)	Fields 5 Records 8
6(b)	Any **two** from: Length check Type check Presence check Format check
6(c)	(see grid below)

Field:	Type	Sold Out	Date	Title
Table:	PERFORMANCE	PERFORMANCE	PERFORMANCE	PERFORMANCE
Sort:				
Show:	□	□	☑	☑
Criteria:	Like "Jazz"	False		
or:				

14.25Summer 2018 P21



6(a)	Fields 5
6(b)	**One** mark description of new code that will allow more than 1000 values **One** mark for example matching candidate's description Example Use a new character instead of N TT345
6(c)	(see grid below)

Field:	At Risk	Age in Years	Type	Map Position
Table:	TREES	TREES	TREES	TREES
Sort:				
Show:	□	□	☑	☑
Criteria:	True	>100		
or:				

Chapter 15

Introduction to Logo

LOGO is a programming language that was developed at the MIT Artificial Intelligence Lab in the 1970s. It was designed to be used as an introduction for people to both programming and artificial intelligence. However, while it is easy to learn, Logo is a powerful language. Once you know the basics, Logo can be used to do extremely complicated things. Logo code has been used in telecommunications, multimedia software and robotics. Finally Logo is FUN!

Turtle

When Logo started, the Turtle was a robotic creature which sat on the floor and was directed to move around by a user typing commands on a computer. Today the turtle is represented by an icon on the screen and can be made to draw on the screen using the same commands. In some environments the turtle looks like a turtle (with head, tail and feet) in others represent the turtle with a triangle.

Logo Commands

The following are some of the most use full commands in the Logo Language, which you will want to become familiar with:

FORWARD - Follow this command with a number (such as: 10 or 1000.) A small number will cause the turtle to move forward a short distance. A larger number will cause it to move further. If you select a large enough number the turtle will go off the canvas and wrap around to the other side.

BACK- Follow this command with a number, the same as FORWARD, only this time the turtle will move backwards.

RIGHT - Follow this command with a number between 0 and 360. The turtle will turn right specified number of degrees.

LEFT - Follow this command with a number between 0 and 360. This command is the same as RIGHT only it will turn the turtle left, not right.

PENUP - This command will cause the turtle to pick up its “pen” up so that you can move the turtle without drawing a line.

PENDOWN - This is the command you would use to put the “pen” back down so you can draw again.

SETPENCOLOR - You can change the colour your turtle draws in. Follow the command with a number to get different colours. For example
“SETPENCOLOR 0” would give you a black pen.

CLEAN- This command will erase the canvas

HOME- This command will move the turtle back to the centre of the canvas

The Repeat Command

You can get your turtle to do one (or several) things repeatedly, without typing them again and again using the REPEAT command. Typing

```
REPEAT 4 [FORWARD 10]
```

Would cause the turtle to move forward 10 spaces, 4 time. So, in total the turtle would move forward 40 spaces. Now Try These. Type:

```
REPEAT 4 [FORWARD 50 RIGHT 90]
```

You should get a square. For a bigger square try replacing 50 with 100 or 200.Type:

```
REPEAT 360 [FORWARD 2 LEFT 1]
```

You should get a circle. For a smaller circle try replacing 2 with 1. Can you make a bigger one?

Your First Program

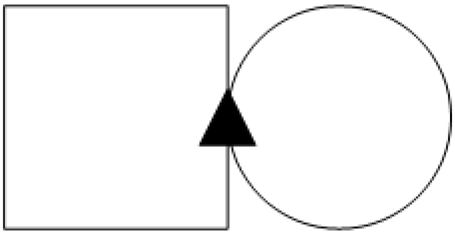
That's a lot to type every time you want to make a square or circle though. Can it be easier? YES. You can teach Logo what a square (or a circle, or a flower) is by making it program. Try typing:

```
TO SQUARE
REPEAT 360 [FORWARD 80 LEFT 90]
END
```

Now type SQUARE and see what happens.

How would you write the program CIRCLE?

How would you write the program CIRCLE_AND_SQUARE to make a drawing that looks like this (where the black triangle is the turtle at the end)?



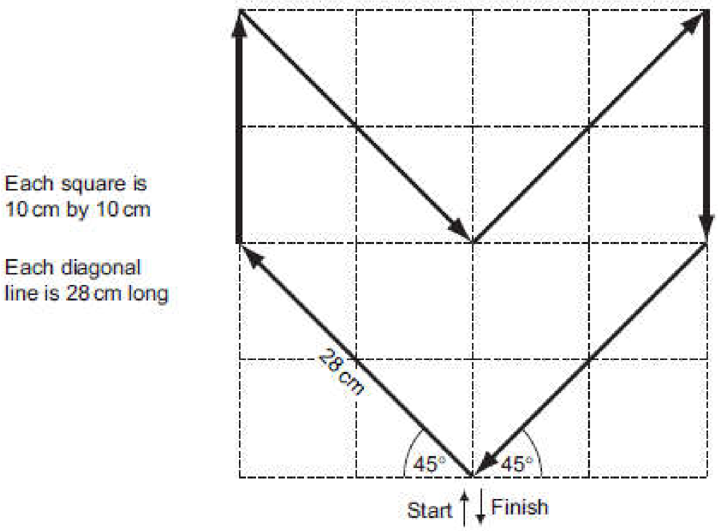
```
TO CIRCLE_AND_SQUARE
    HOME
    CLEAN
    CIRCLE
    FORWARD 52
    REPEAT 3 [ LEFT 90 FORWARD 104]
    LEFT 90
    FORWARD 52
    END
```

CIRCLE was defined as:

```
TO CIRCLE
REPEAT 360 [FORWARD 1 RIGHT 1]
END
```

15.1 Summer 2014 P11

A floor turtle can use the following instructions.



Complete the set of instructions to draw the above shape.

Pen Down ……………………………………………………………………………

Left 45 ……………………………………………………………………………

……………………………………………………………………………

……………………………………………………………………………

……………………………………………………………………………

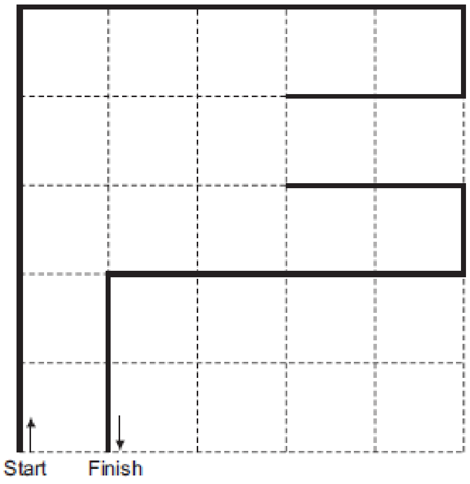
……………………………………………………………………………

……………………………………………………………………………

15.2 Summer 2014

A floor turtle uses the following commands:

In the following grid, each of the squares measures 10 cm by 10 cm:



Complete the set of instructions to draw the shape shown above (in bold lines).

Pen Down ……………………………………………………………………………

Repeat 2……………………………………………………………………………

……………………………………………………………………………

……………………………………………………………………………

……………………………………………………………………………

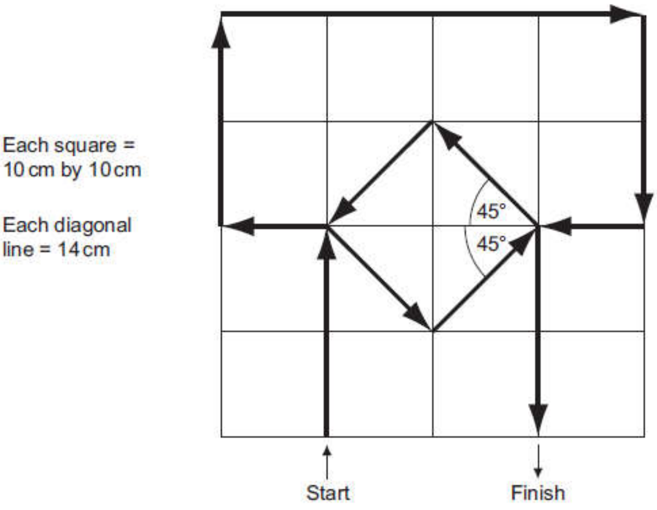
....
....
....

1 mark for each block of code between dotted lines. (e.g. use of REPEAT and BACKWARD functions). If an error occurs in the code, try to find a correct code sequence later on in the answer (in cases such as this, it is often easier to work backwards from last statement looking for correct blocks).

15.3 Summer 2012

A floor turtle can use the following instructions.

Instruction	Meaning
FORWARD ***x***	Move ***x*** cm forwards
LEFT ***t***	Turn left ***t*** degrees
RIGHT ***t***	Turn right ***t*** degrees
REPEAT ***n***	Repeat next set of instructions ***n*** times
ENDREPEAT	Finish repeated instructions
PENUP	Lift the pen
PENDOWN	Lower the pen



Complete the set of instructions to draw the above shape.

Pen Down ....

Forward 20...

Left 90 ....

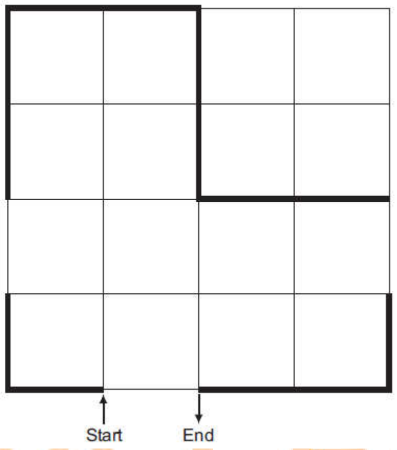
....
....
....

15.4 Summer 2010

A floor turtle can use the following instructions:

Instruction	Meaning
FORWARD ***d***	Move ***d*** cm forward
BACKWARD ***d***	Move ***d*** cm backward
LEFT ***t***	Turn left ***t*** degrees
RIGHT ***t***	Turn right ***t*** degrees
REPEAT ***n***	Repeat the next set of instructions ***n*** times
ENDREPEAT	End of REPEAT loop
PENUP	Raise the pen
PENDOWN	Lower the pen

(In the following grid, each square is 10 cm by 10 cm.)



Complete the set of instructions to draw the above shape.

Pen Down ....

Left 90 ....

Forward 10..

Right 90 ....

....

....

....

....

Marking Scheme

15.1 Summer 2014 P11

pendown
left 45
forward 28
right 45 (1 mark)
forward 20
right 135 (1 mark)
forward 28
left 90 (1 mark)
forward 28
right 135 (1 mark)
forward 20
right 45 (1 mark)
forward 28
(penup)
(left 45)

15.2 Summer 2014

PENDOWN
REPEAT 2
3 FORWARD 50 FORWARD 25 ENDREPEAT
4 RIGHT 90 RIGHT 90 FORWARD 50
5 ENDREPEAT RIGHT 90

- -

6 FORWARD 10
7 RIGHT 90
8 FORWARD 20

- - - - - - - - - - - - - - - -

9 PENUP (statements 9 and 10 are interchangeable)
10 LEFT 90
11 FORWARD 10

- - - - - - - - - - - - - - - -

12 PENDOWN
13 LEFT 90 (statements 12 and 13 are interchangeable)
14 FORWARD 20
15 RIGHT 90

- - - - - - - - - - - - - - - -

16 FORWARD 10
17 RIGHT 90
18 FORWARD 40

- - - - - - - - - - - - - - - -

19 LEFT 90
20 FORWARD 20
(21 PENUP) (line 21 is not essential) [6

15.3 Summer 2012

pendown
forward 20
left 90

- - - - - - - - - - - -

forward 10
right 90 (1 mark)
forward 20

- - - - - - - - - - - -

right 90
forward 40
right 90 (1 mark)
forward 20
right 90

- - - - - - - - - - - -

forward 10
right 45 (1 mark)
forward 14

- - - - - - - - - - - -

repeat 3 or left 90
left 90 or forward 14
forward 14 or left 90
endrepeat or forward 14 (1 mark)
left 90
forward 14

- - - - - - - - - - - -

right 135
forward 20 (1 mark)
(PENUP)

15.4 Summer 2010

LEFT 90
PENDOWN
FORWARD 10
RIGHT 90

————————————

FORWARD 10
PENUP
FORWARD 10
PENDOWN

————————————

FORWARD 20
RIGHT 90
FORWARD 20
RIGHT 90
FORWARD 20

————————————

LEFT 90
FORWARD 20
PENUP / RIGHT 90

————————————

20 RIGHT 90/PENUP
FORWARD 10
PENDOWN

————————————

FORWARD 10
RIGHT 90
FORWARD

Chapter 16

2.2 Programming

Visual Basic .NET (VB.NET) is an object-oriented computer programming language implemented on the .NET Framework. Although it is an evolution of classic Visual Basic language, it is not backwards-compatible with VB6, and any code written in the old version does not compile under VB.NET. Like all other .NET languages, VB.NET has complete support for object-oriented concepts. Everything in VB.NET is an object, including all of the primitive types (Short, Integer, Long, String, Boolean, etc.) and user defined types, events, and even assemblies. All objects inherit from the base class Object.

VB.NET is implemented of Microsoft's .NET framework. Therefore it has full access all the libraries in the .Net Framework. It's also possible to run VB.NET programs on Mono, the open-source alternative to .NET, not only under Windows, but even Linux or Mac OSX.

The following reasons make VB.Net a widely used professional language:

- Modern, general purpose.
- Object oriented.
- Component oriented.
- Easy to learn.
- Structured language.
- It produces efficient programs.
- It can be compiled on a variety of computer platforms.
- Part of .Net Framework.
- Strong Programming Features VB.Net

VB.Net has numerous strong programming features that make it endearing to multitude of programmers worldwide. Let us mention some of these features:

- Boolean Conditions
- Automatic Garbage Collection
- Standard Library
- Assembly Versioning
- Properties and Events
- Delegates and Events Management
- Easy to use Generics
- Indexers
- Conditional Compilation
- Simple Multithreading

VB.NET – ENVIRONMENT

Integrated Development Environment (IDE) For VB.Net

Microsoft provides the following development tools for VB.Net programming:

- Visual Studio 2010 (VS)
- Visual Basic 2010 Express (VBE)
- Visual Web Developer

The last two are free. Using these tools you can write all kinds of VB.Net programs from simple command-line applications to more complex applications. Visual Basic Express and Visual Web Developer Express edition are trimmed down versions of Visual Studio and has the same look and feel. They retain most features of Visual Studio. In this tutorial, we have used Visual Basic 2010 Express and Visual Web Developer

Download and Setup

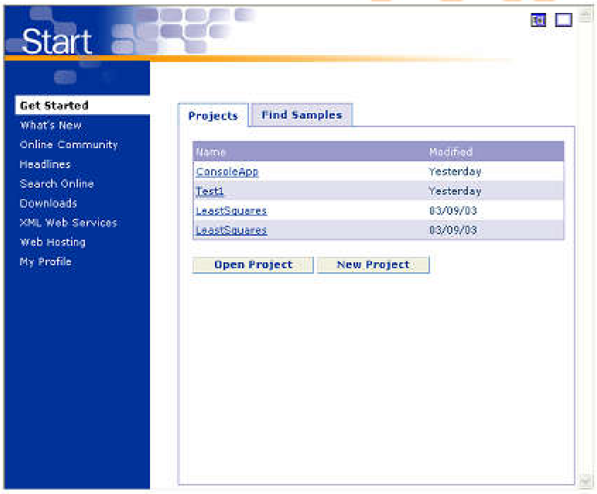
Download VB Express and install it on your computer at home.

http://www.microsoft.com/visualstudio/eng/products/visual-studio-2010-express

Console Application

A VB.NET console application uses a command line window for its user interface. Because there is no opportunity for the user to provide inputs other than those asked for by the program, a console application is not “event driven”. The actions performed in the program must follow in a linear fashion, and are completely defined by the program, and thus the programmer. This “procedural” programming is limiting, but makes a good introduction since the complexity of the user interface is eliminated.

1. Start Microsoft Visual Studio.NET. Note that it is not listed as Visual Basic.NET. Visual Studio.NET supports other programming languages other than VB. A Start Screen is displayed to help you open new or existing VB projects.

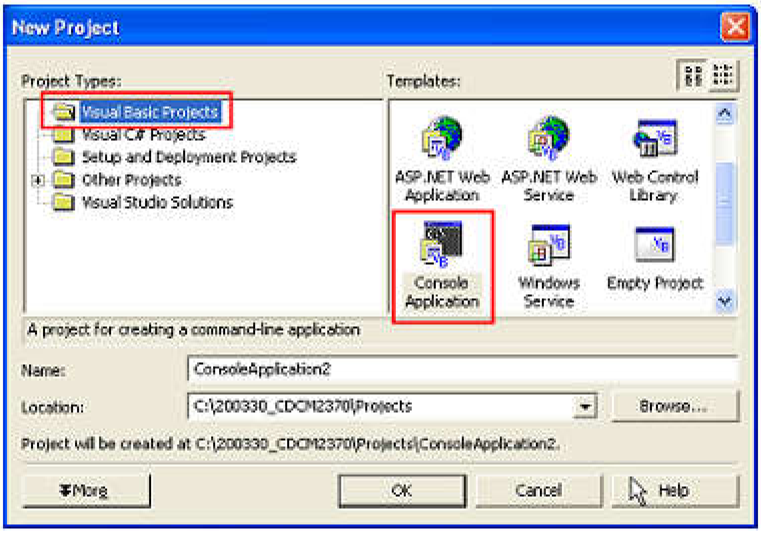


Name	Modified
ConsoleApp	Yesterday
Test1	Yesterday
LeastSquares	03/09/03
LeastSquares	03/09/03

2. Click the New Project button.

3. The New Project dialog box is shown. You select the type of project and name it here.

4. Ensure that the Visual Basic projects folder is opened, and then scroll to and click Console Application from the list of templates.



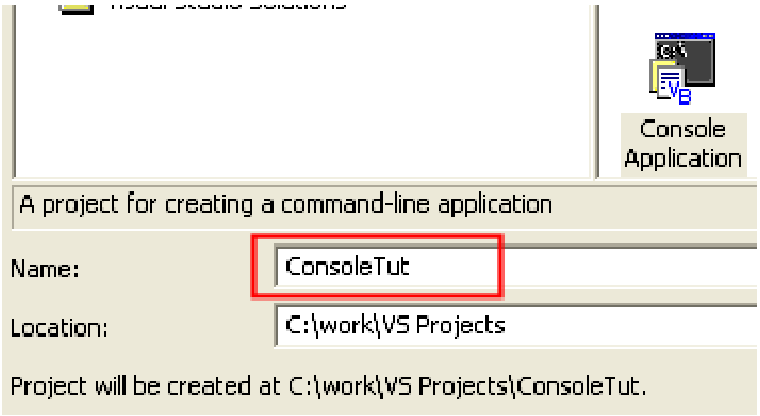
A project for creating a command-line application

Name: ConsoleApplication2

Location: C:\200330_CDCM2370\Projects

Project will be created at C:\200330_CDCM2370\Projects\ConsoleApplication2.

5. Enter ConsoleTut as the name of the project. All the files associated with the project will be stored in a folder as indicated in the Location text box (note this so you can find it later). Click OK to create the project.



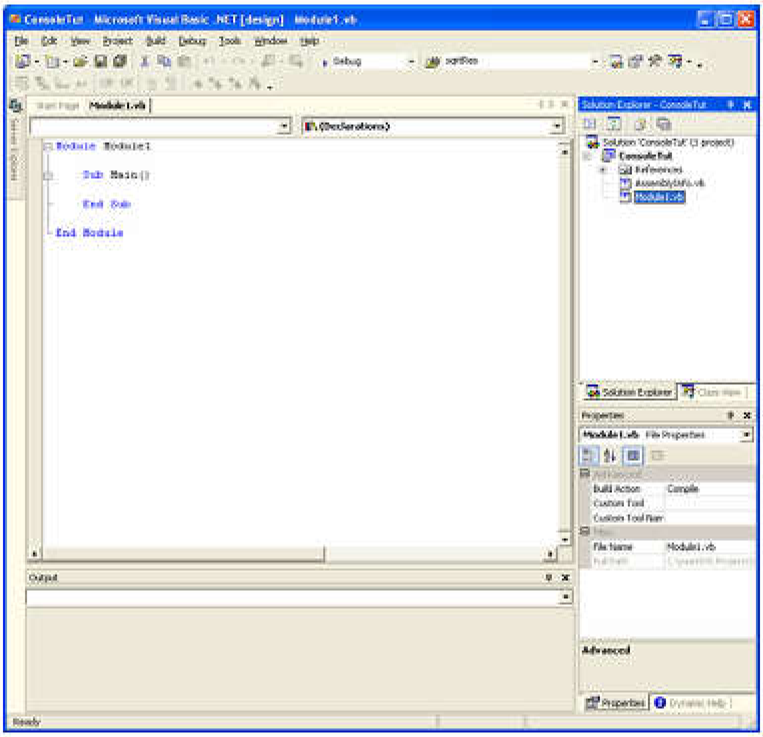
A project for creating a command-line application

Name: ConsoleTut

Location: C:\work\VS Projects

Project will be created at C:\work\VS Projects\ConsoleTut.

6. The VS (Visual Studio) IDE (Integrated Development Environment) is where you build your program. You can actually create any VB program in Notepad, but the IDE provides a large number of tools that greatly simplifies the process of creating a Windows program.



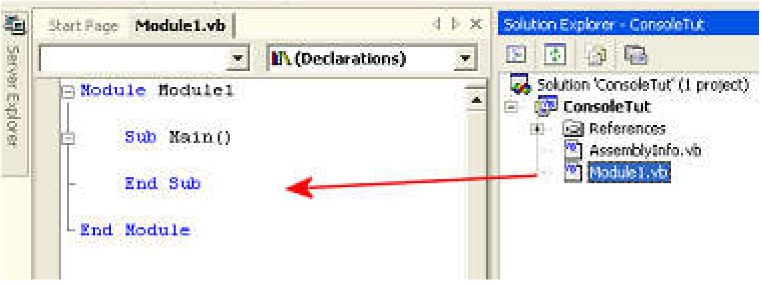
```
Module Module1

    Sub Main()

    End Sub

End Module
```

7. For this tutorial, you can ignore most of the windows and tools in the IDE. The Solution Explorer in the top right corner of the IDE shows the files associated with the project. The console application template provides the required files. The Module1.vb file is the active file, and its contents are shown in the editor window.



```
Module Module1

    Sub Main()

    End Sub

End Module
```

8. The code contained within the Module1.vb file is listed between the two statements:

Module Module1

End Module

The module currently contains a single subroutine (don't worry too much about terms just yet) called Main. The code inside the Main subroutine is automatically "run" when the program starts. You will add all the code for this tutorial in between the Sub Main() and End Sub statements.

Tools: (Console.ReadLine for Input, Console.WriteLine for Output)

The console application simplifies input and output in that the user interface is essentially text based. The user types in the input, and the output is posted to the same text window. This text window is referenced in your program code by using the keyword Console. Three “actions” associated with the Console object enable you to receive input (ReadLine) and output (Write and WriteLine) text characters to the console window.

Basic data types

In order for a computer system to process and store data effectively, different kinds of data are formally given different types. This enables:

- data to be stored in an appropriate way, for example, as numbers or characters
- data to be manipulated effectively, for example numbers with mathematical operators and characters with concatenation
- automatic validation in some cases.

INTEGER

An **INTEGER** is a positive or negative whole number that can be used with mathematical operators.

SINGLE

A **Single** is a positive or negative number with a fractional part. Single numbers can be used with mathematical operators. (In pseudocode REAL data type is used for numbers with fractional part)

CHAR

A variable or constant of type **CHAR** is a single character.

Gender = 'F'

STRING

A variable or constant of type **STRING** is several characters in length. Strings vary in length and may even have no characters: an empty string. The characters can be letters and/or digits and/or any other printable symbol.

BOOLEAN

A **BOOLEAN** variable can have only two values: TRUE or FALSE.

Variables:

A variable is a named location in the computer’s memory that stores a certain type of data (character, string of characters, integer, real number, and so on). The variable is identified in the program by a unique name.

The values stored in any variable are changed during execution of program.

By storing each of the inputs in memory (in a variable), you can use them again and again within the program without requiring that the user re-input them.

Variable Declaration:

Before variables can be used to store data, they must be “Declared”. These reserves the appropriate amount of memory to store the value assigned to the variable.
In the editor window, below Sub Main() but above End Sub, type:

```
DIM Name As String
DIM Weight1, Weight2, Weight_Difference  As Single
DIM Count As Integer
```

Constant:

A constant is a memory location in which stored values remain unchanged during execution of program.

Constant Declaration and Initialization:

Before constants can be used, they must be declared and initialized with permanent values. Declaration and assignment of permanent value is done in the same line like:

```
ConstTaxRate as single=0.15
ConstMaxMarks as single=100
```

Array:

Data stored in a computer is stored at any location in memory that the computer decides to use, which means that similar pieces of data can be scattered all over memory. This, in itself, doesn't matter to the user, except that to find each piece of data it has to be referred to by a name (known as a "variable name"). For example, if we want to store the 20 names of students in a group then each location would have to be given a different variable name. The first, "Abdullah", might be stored in location Name1, the second, "Rumaisa", might be stored inName2 , the third, "Rashid", could be stored in Name3.Creating the 20 different variable names is possible but these names are separate (as far as the computer is concerned) and do not relate to each other in any way. It would be far more sensible to force the computer to store them all together using the same variable name. However, this doesn't let me identify individual names. If I call the first one, Name(1) , the second one Name(2) and so on, it is obvious that they are all people's names and that they are distinguishable by their position in the list. A list like this is called an **array.**

Each element in the array is identified using its **subscript** or **index number**. The largest and smallest index numbers are called the *upper bound* and *lower bound* of the array.

	Name
1	Abdullah
2	Rumaisa
3	Rashid
4	Afeera
5	Laiba
6	Patel
7	Smith
19	Mani
20	Muzna

Declaring an array

It is important declare the arrays before assigning values in it so that program can reserve that amount of space in its memory; otherwise, there may not be enough space when the program uses the data.

Declaration consists of telling the computer program:

- the identifier name of the array
- the sort of data that is going to be stored in the array, i.e. its data type
- Howmany items of data are going to be stored, so that it knows how much space to reserve.

Different programming languages have different statements for initialising the array but they all do the same thing. In Visual Basic, the statement is:

```
Dim Name(20) As String
```

This Dim statement declares:

- the identifier name: Name
- the upper bound: 20
- the data type: String.

The upper bound of 20 specifies that there can be a maximum of 21 data items, since Visual Basic starts with a subscript of zero. We do not have to fill the array; the upper bound of 20 indicates the maximum size.

The array that has been described in one dimension array so far is really only a list of single data items. It is possible to have an array which can be visualised as a two-dimensional table with rows and columns and a data value in each cell.

Reading data into an array

To assign data values to the elements of the array, we do this with assignment statements such as:

Name(6) = “Patel”

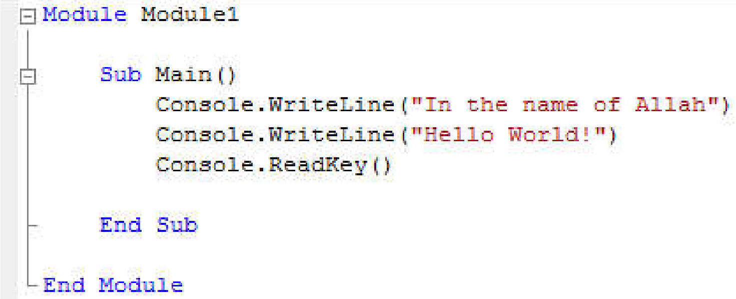
This places the string “Patel” at index position 6 in the array.

Similarly, the following statement places the string “Rashid” at index position 3 in the array.

Name(19) = “Mani”

First program

Start a new Project and select Console Application. Type the following code:

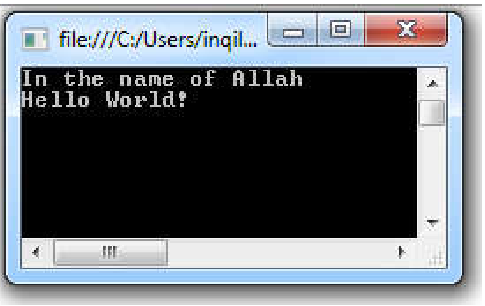


```
Module Module1

    Sub Main()
        Console.WriteLine("In the name of Allah")
        Console.WriteLine("Hello World!")
        Console.ReadKey()

    End Sub

End Module
```



```
In the name of Allah
Hello World!
```

(Press F5 to run the code)

Tasks

0.1 – Write a program that displays the message “Asslam-o-Alaikum”

0.2 – Write a program that displays the message “My name is Muhammad and I live in Brussels” (replace Muhammad and Brussels with your own information)

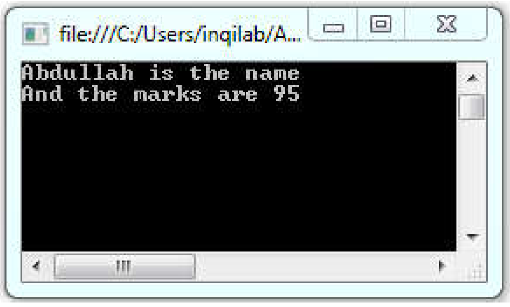
Example Program 1 - Assignment - *Integer, byte, real, boolean, character, string, date/time.*

```
Module Module1
Sub Main()
Dim Marks AsSingle
Dim Name AsString

        Name = "Abdullah"
        Marks = 95
        Console.WriteLine(Name &" is the name")
        Console.WriteLine("And the marks are "& Marks)
        Console.ReadKey()

        Name = "Muzna"
        Console.Write("Enter marks:    ")
        Marks = Console.ReadLine()
        Console.WriteLine("Now "& Name &" is the name and the marks are  "&
Marks)

        Console.ReadKey()
EndSub
EndModule
```



```
Abdullah is the name
And the marks are 95
```

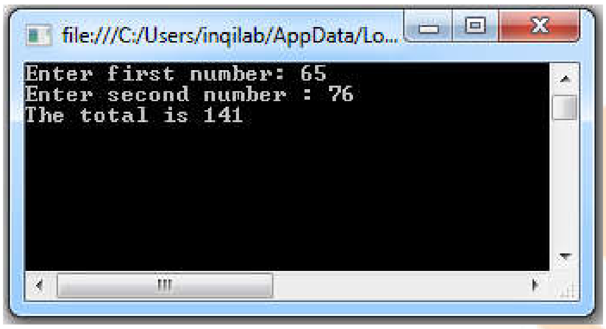
Example Program 2 - Arithmetic - +, –, /, x, DIV, MOD

```
Module Module1
Sub Main()
Dim Number1, Number2, Total AsSingle

        Console.Write("Enter first number:  ")
        Number1 = Console.ReadLine()
        Console.Write("Enter second number : ")
        Number2 = Console.ReadLine()
        Total = Number1 + Number2
        Console.WriteLine("The total is "& Total)

        Console.ReadKey()

EndSub
EndModule
```



```
Enter first number: 65
Enter second number : 76
The total is 141
```

Tasks

3.1) Write a program that divides a number entered by the user by 2

3.2) Write a program that displays the 7 times table

3.3) Write a program that displays any times table the user requests

Example Program 3 – Selection

```
Module Module1

Sub Main()
Dim Marks AsSingle

        Console.Write("Enter marks……    ")
        Marks = Console.ReadLine()
If Marks >= 50 Then
            Console.WriteLine("Congrats! You are pass.")
Else
            Console.WriteLine("Sorry! You are fail.")
EndIf
        Console.ReadKey()

EndSub

EndModule
```

Tasks

1. Write a program to ask the user what 24+9 is. Say “Excellent” if they get it right.
2. Write a program to ask the user “how many in a bakers dozen?” and say “mostexcellent” if they get it right.
3. Write a program to ask the user to enter their age. If their age is under 18 then say“Sorry, you are not allowed to vote”.

4. Write a program to ask the user for two numbers. Compare the first with the second and then print out one of three messages. Either the numbers are equal, the first isbigger, or the second is bigger. You will need more than one IF to solve this one.
5. Write a program which asks the user to enter their password. If they enter the word“PASSWORD” then display the message “Welcome to the treasure”, otherwisedisplay a message which says “go away, it’s all mine”.
6. Write a program which asks the user to enter a number between 1 and 10. If thenumber entered is out with this range then display a message “Sorry…out of range”.

Example Program 4 - Relational operators - =, <, >, <>, <=, >=

```
Module Module1

Sub Main()
Dim Age AsInteger
        Console.WriteLine("What is your age?")
        age = Int(Console.ReadLine())
If Age >= 18 Then
            Console.WriteLine("You can drive.")
Else
            Console.WriteLine("You are too young to drive")
EndIf
        Console.ReadKey()
EndSub
      EndModule
```

The symbols we can use to test for conditions are as follows:

<	Less than
<=	Less Than or Equal To
>	Greater than
>=	Greater Than or Equal To
==	IS Equal To
!= or <>	Not Equal To

Example Program 5 - Boolean operators - *NOT, AND, OR*

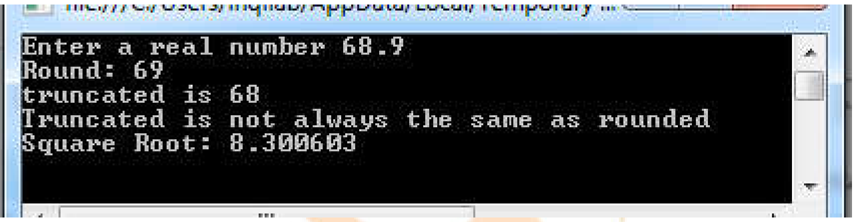
```
Module Module1
Sub Main()
Dim Age, Points AsInteger
        Console.Write("What is your age? ")
        Age = Console.ReadLine()
        Console.Write("How many points do you have on your licence? ")
        Points = Console.ReadLine()
If Age > 16 And Points < 9 Then
            Console.WriteLine("You can drive!")
Else
            Console.WriteLine("You are not eligable for a driving licence")
EndIf
        Console.ReadKey()
EndSub
EndModule
```

Example Program 6 - Built-in functions- *Arithmetic functions: round, truncation.*

```
Module Module1
Sub Main()
Dim Num, Rounded, Trunc, SquareRoot AsSingle

        Console.Write("Enter a real number ")
        num = Console.ReadLine()
        rounded = Math.Round(num)
        Trunc = Math.Truncate(Num)
        SquareRoot = Math.Sqrt(Num)
        Console.WriteLine("Round: "& Rounded & vbNewLine &"truncated is "& Trunc)
        Console.WriteLine("Truncated is not always the same as rounded")
        Console.WriteLine("Square Root: "& SquareRoot)

        Console.ReadKey()
EndSub
EndModule
```



```
Enter a real number 68.9
Round: 69
truncated is 68
Truncated is not always the same as rounded
Square Root: 8.300603
```

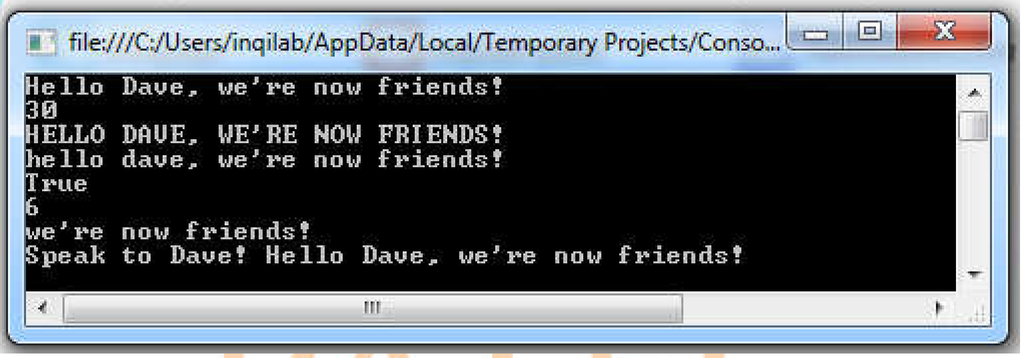
Tasks

1) Write a program that asks for 5 numbers, calculates the mean average and then rounds it down. Display the result on screen.

Example Program 7 - String handling functions *length, position, substring, concatenation*.

```
Module Module1
Sub Main()
Dim theString AsString
        theString = "Hello Dave, we're now friends!"
        Console.WriteLine(theString)
        Console.WriteLine(theString.Length) 'display the string's length
        Console.WriteLine(theString.ToUpper) 'display the string in upper case
        Console.WriteLine(theString.ToLower) 'display the string in lower case
        Console.WriteLine(theString.Contains("Dave")) 'is Dave there?
        Console.WriteLine(theString.IndexOf("D")) 'position of D
        Console.WriteLine(theString.Substring(12)) 'displays the substring
starting at position(12)
Dim newString AsString
        newString = "Speak to Dave! "& theString 'string concatenation
        Console.WriteLine(newString)
        Console.ReadKey() ' pause and wait so user can read output.

        Console.ReadKey()
EndSub
EndModule
```

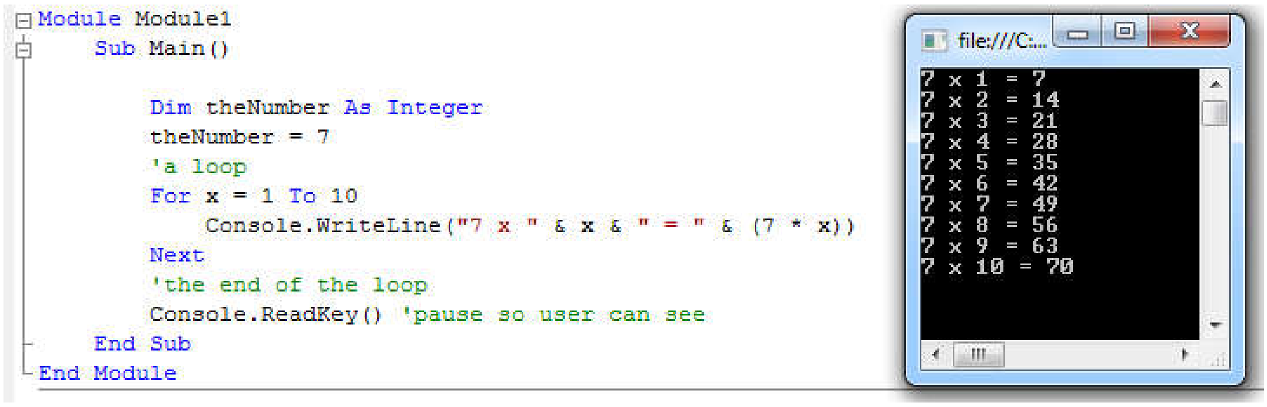

file:///C:/Users/inqilab/AppData/Local/Temporary Projects/Conso...

```
Hello Dave, we're now friends!
30
HELLO DAVE, WE'RE NOW FRIENDS!
hello dave, we're now friends!
True
6
we're now friends!
Speak to Dave! Hello Dave, we're now friends!
```

Tasks

1. Write a program that checks a username against a stored value. How the user enters the username should NOT be case sensitive.
2. Adapt program 1 so that it also takes in a password. If the user enters spaces after the password the computer will trim them out automatically.
3. Write a program that will check a phone number is of the correct length.
4. Write a program that asks for a user's full name in one inputbox/textbox but then stores the first and second names in different variables.

Example Program 8 – Repetition



```
Module Module1
    Sub Main()

        Dim theNumber As Integer
        theNumber = 7
        'a loop
        For x = 1 To 10
            Console.WriteLine("7 x " & x & " = " & (7 * x))
        Next
        'the end of the loop
        Console.ReadKey() 'pause so user can see
    End Sub
End Module
```

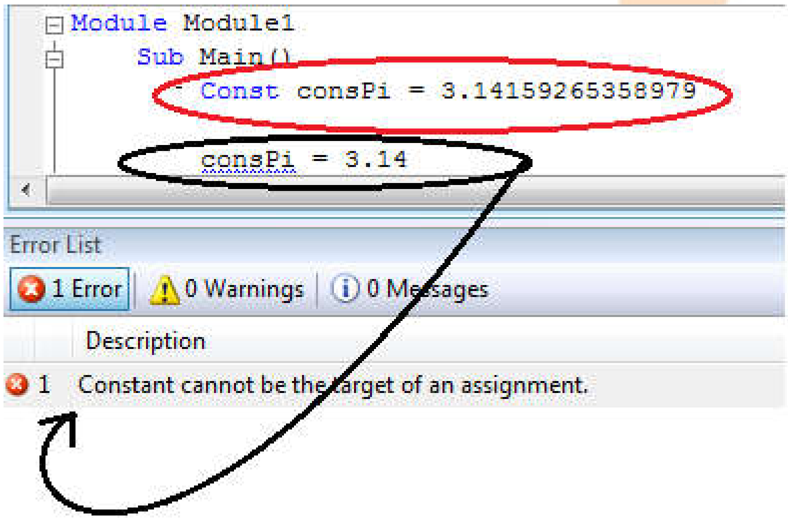
```
7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
7 x 10 = 70
```

Tasks

1. Write a program which asks for your name and then displays it 5 times on the screen.
2. Write a program to display the name of the town you live in 10 times.
3. Write a program to ask for a person's favourite CD and the artist. Both should be displayed on the same line 5 times.
4. Write a program to ask for a number and display its multiplication table 1 to 100
5. Write a program that asks the user for a number 5 times and adds them all up to give a total.

Example Program 9 - Constants

'a constant is a value that doesn't change
'using them greatly improves the readability of your code
'number constants

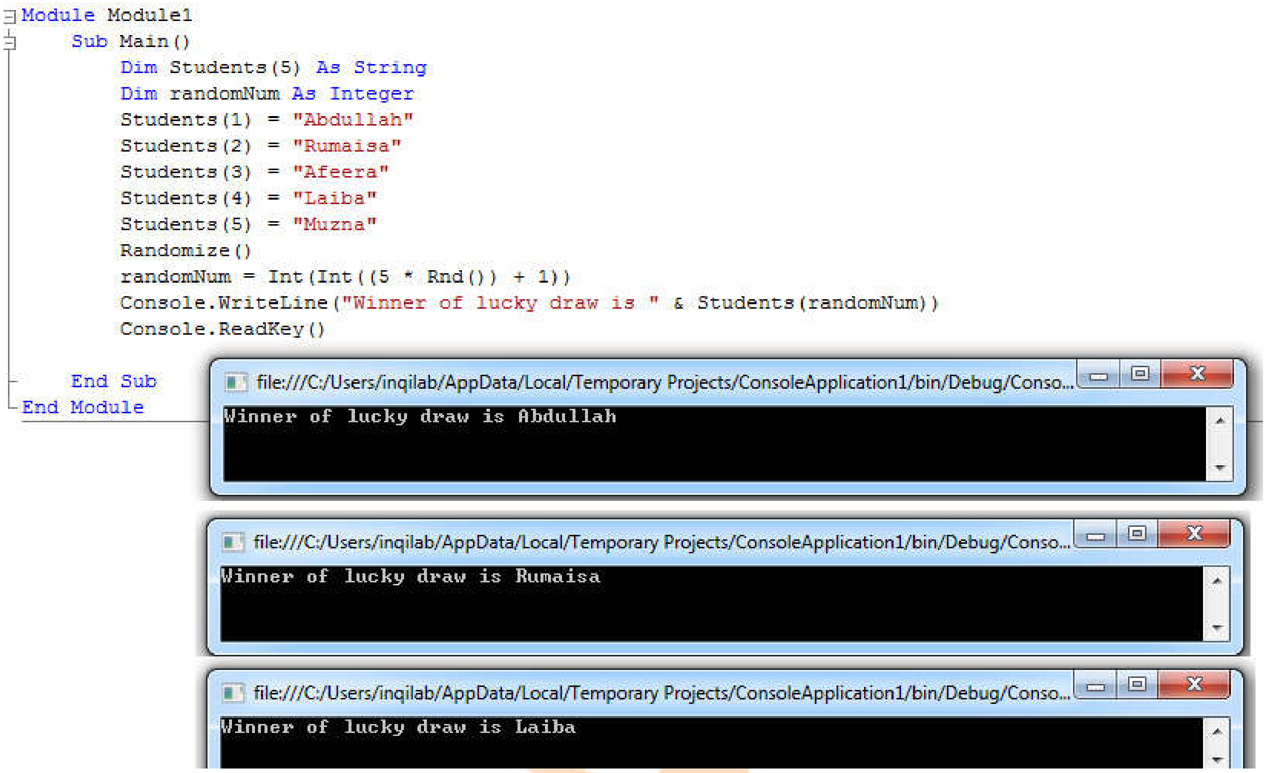


```
Module Module1
    Sub Main()
        Const consPi = 3.14159265358979

        consPi = 3.14
```

1 Constant cannot be the target of an assignment.

Example Program 10 – 1 dimensional arrays



```
Module Module1
    Sub Main()
        Dim Students(5) As String
        Dim randomNum As Integer
        Students(1) = "Abdullah"
        Students(2) = "Rumaisa"
        Students(3) = "Afeera"
        Students(4) = "Laiba"
        Students(5) = "Muzna"
        Randomize()
        randomNum = Int(Int((5 * Rnd()) + 1))
        Console.WriteLine("Winner of lucky draw is " & Students(randomNum))
        Console.ReadKey()

    End Sub
End Module
```

file:///C:/Users/inqilab/AppData/Local/Temporary Projects/ConsoleApplication1/bin/Debug/Conso...

```
Winner of lucky draw is Abdullah
```

file:///C:/Users/inqilab/AppData/Local/Temporary Projects/ConsoleApplication1/bin/Debug/Conso...

```
Winner of lucky draw is Rumaisa
```

file:///C:/Users/inqilab/AppData/Local/Temporary Projects/ConsoleApplication1/bin/Debug/Conso...

```
Winner of lucky draw is Laiba
```

Tasks

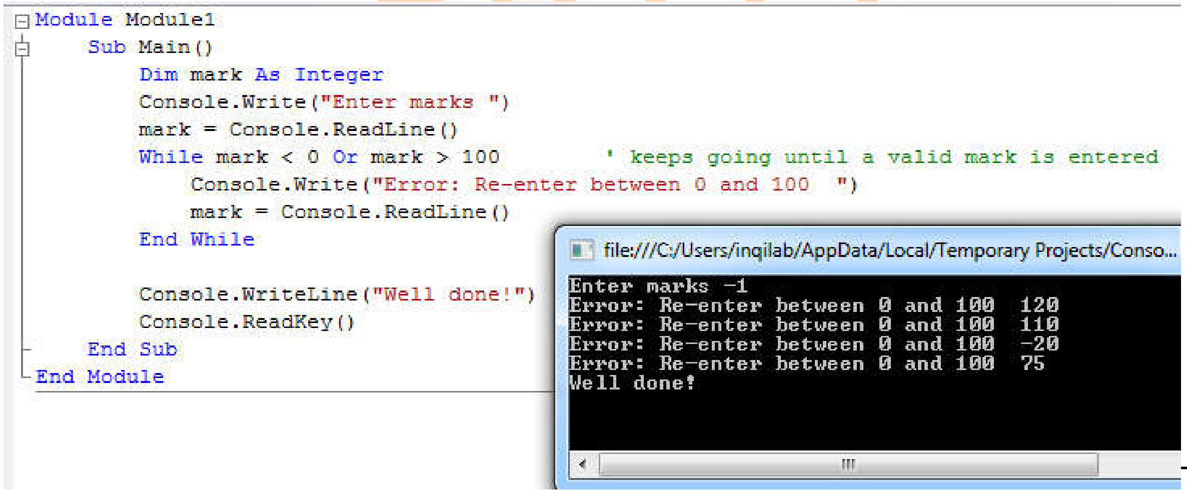
1) Write a program which will set up an array to hold 50 numbers. Call the array numbers. Displaythe array's contents across the screen. They should all be 0.

2) Create a program that stores an array of car records. At least 5 cars and 4 fields per record.

3) Create a program that stores an array of 5 people records. The information should be entered bythe user.

4) Adapt program 2 to now do a linear search for a certain car and display it’s details.

Example Program 11 - Validation



```
Module Module1
    Sub Main()
        Dim mark As Integer
        Console.Write("Enter marks ")
        mark = Console.ReadLine()
        While mark < 0 Or mark > 100          ' keeps going until a valid mark is entered
            Console.Write("Error: Re-enter between 0 and 100  ")
            mark = Console.ReadLine()
        End While

        Console.WriteLine("Well done!")
        Console.ReadKey()
    End Sub
End Module
```

file:///C:/Users/inqilab/AppData/Local/Temporary Projects/Conso...

```
Enter marks -1
Error: Re-enter between 0 and 100  120
Error: Re-enter between 0 and 100  110
Error: Re-enter between 0 and 100  -20
Error: Re-enter between 0 and 100  75
Well done!
```

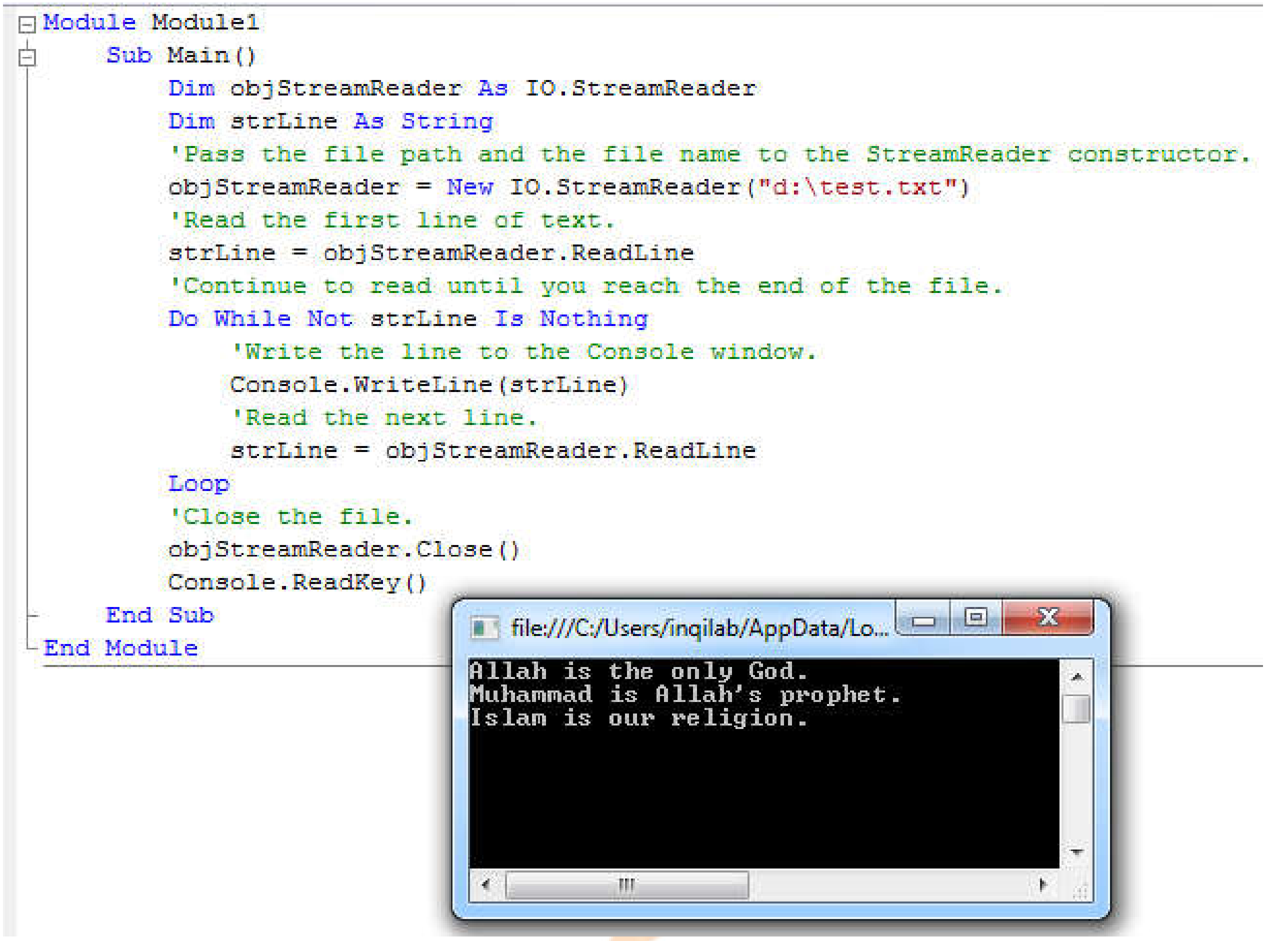
Tasks

1) Write a program that validates a user is old enough to drive (older than 17, younger than 80)

2) Write a program that checks that a telephone number entered is long enough (string length)

3) Write a program that checks that both a username and password is correct before allowing you to proceed.

Example Program 12 – Read from a text file



```
Module Module1
    Sub Main()
        Dim objStreamReader As IO.StreamReader
        Dim strLine As String
        'Pass the file path and the file name to the StreamReader constructor.
        objStreamReader = New IO.StreamReader("d:\test.txt")
        'Read the first line of text.
        strLine = objStreamReader.ReadLine
        'Continue to read until you reach the end of the file.
        Do While Not strLine Is Nothing
            'Write the line to the Console window.
            Console.WriteLine(strLine)
            'Read the next line.
            strLine = objStreamReader.ReadLine
        Loop
        'Close the file.
        objStreamReader.Close()
        Console.ReadKey()
    End Sub
End Module
```

```
Allah is the only God.
Muhammad is Allah's prophet.
Islam is our religion.
```

Tasks

1) Write a program that reads the students' names from a txt file and displays them on the screen

2) Write a program that reads 10 team names from a txt file and stores them in an array

3) Write a program that reads 5 song titles from a csv file and displays them on the screen

4) Write a program that reads 20 team names from a csv file into an array, then displays the array on screen

Past Paper Questions

Q 1a) Write an algorithm, using pseudo code and a FOR … TO … NEXT loop structure, to input 1000 numbers into an array.

....

....

....

....

....

....

....

....

....

....

....

....

....

b) Rewrite your algorithm using another loop structure.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

Q 2 (a) Write an algorithm, using pseudo code or flowchart only, which:

• inputs three numbers

• outputs the largest of the three numbers

....

....

....

....

....

....

....

....

....

....

....

....

....

....

b) Write an algorithm, using pseudo code or flowchart only, which:

• inputs 1000 numbers

• outputs how many of these numbers were whole numbers (integers)

(You may use INT(x) in your answer, e.g. y = INT(3.8) gives the value y = 3)

....

....

....

....

....

....

....

....

....

....

....

Q 3) Write an algorithm, using pseudo code, to input a number between 0 and 100 inclusive. The algorithm should prompt for the input and output an error message if the number is outside this range.

....

....

....

....

....

....

....

....

....

....

....

Q 4) Rewrite the following pseudo code algorithm using a WHILE … DO … ENDWHILE loop.

```
      Dim Num, A(12) AsSingle
Dim Counter AsInteger
        Console.Write("Enter a number ")
        Num = Console.ReadLine
For Counter = 1 To 12
            Num = Num * Counter
            A(Counter) = Num
Next
For Counter = 1 To 12
            Console.WriteLine(A(Counter))
Next
```

....

....

....

....

....

....

....

....

....

....

....

Q 5 (a) Write an algorithm to input three different numbers, and then output the largest number. Use **programming statements**.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

(b) Write an algorithm in programming, using a single loop, to print 50 names that have been stored in an array.

....

....

....

....

....

....

....

....

....

....

....

....

....

Q 6) Write an algorithm using programming language, to:

- input a positive integer
- use this value to set up how many other numbers are to be input
- input these numbers
- calculate and output the total and the average of these numbers.

....

....

....

....

....

....

....

....

....

....

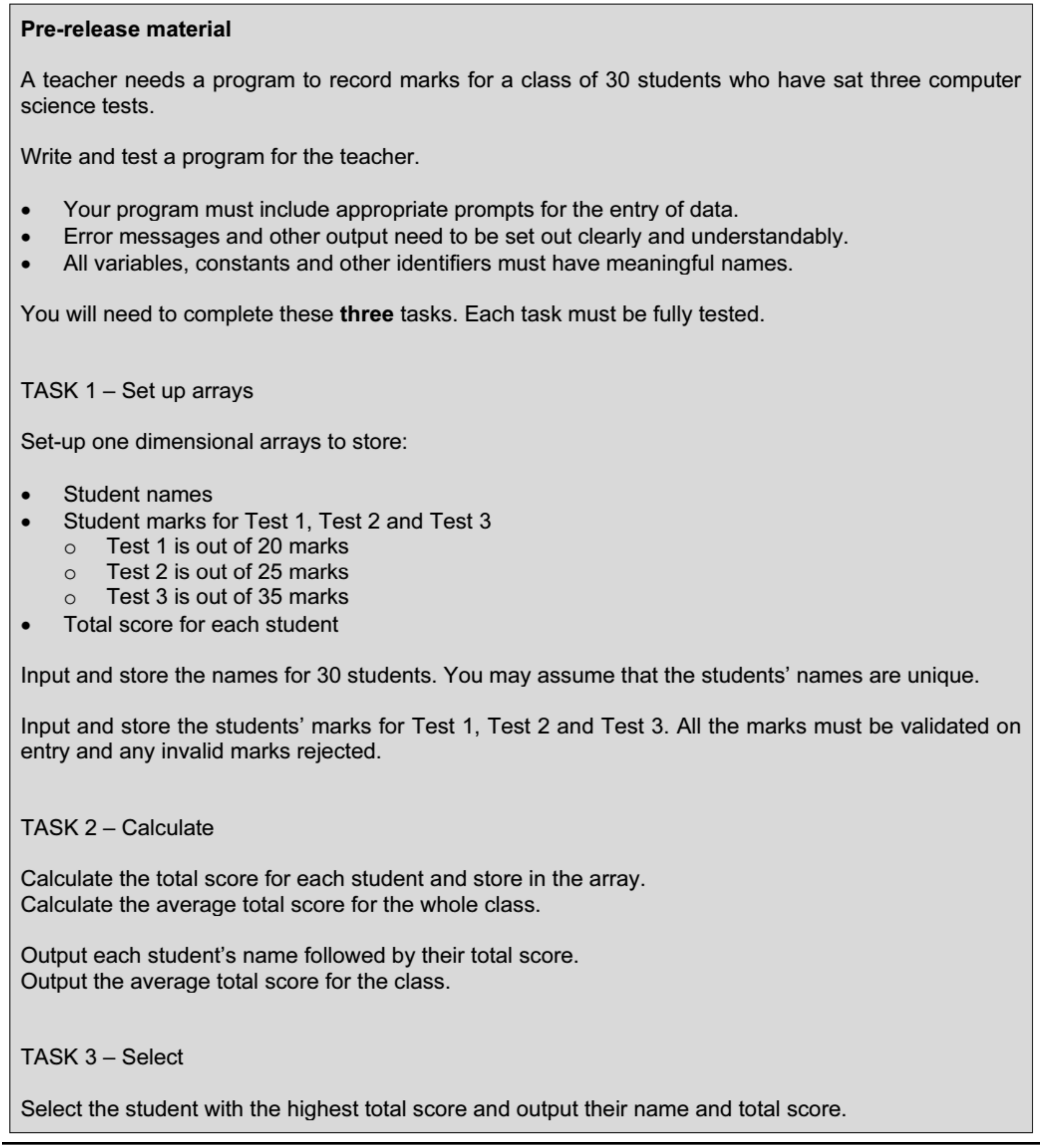
....

....

....

....

Pre-Release Materials



Pre-release material

A teacher needs a program to record marks for a class of 30 students who have sat three computer science tests.

Write and test a program for the teacher.

- Your program must include appropriate prompts for the entry of data.
- Error messages and other output need to be set out clearly and understandably.
- All variables, constants and other identifiers must have meaningful names.

You will need to complete these **three** tasks. Each task must be fully tested.

TASK 1 – Set up arrays

Set-up one dimensional arrays to store:

- Student names
- Student marks for Test 1, Test 2 and Test 3
    - Test 1 is out of 20 marks
    - Test 2 is out of 25 marks
    - Test 3 is out of 35 marks
- Total score for each student

Input and store the names for 30 students. You may assume that the students' names are unique.

Input and store the students' marks for Test 1, Test 2 and Test 3. All the marks must be validated on entry and any invalid marks rejected.

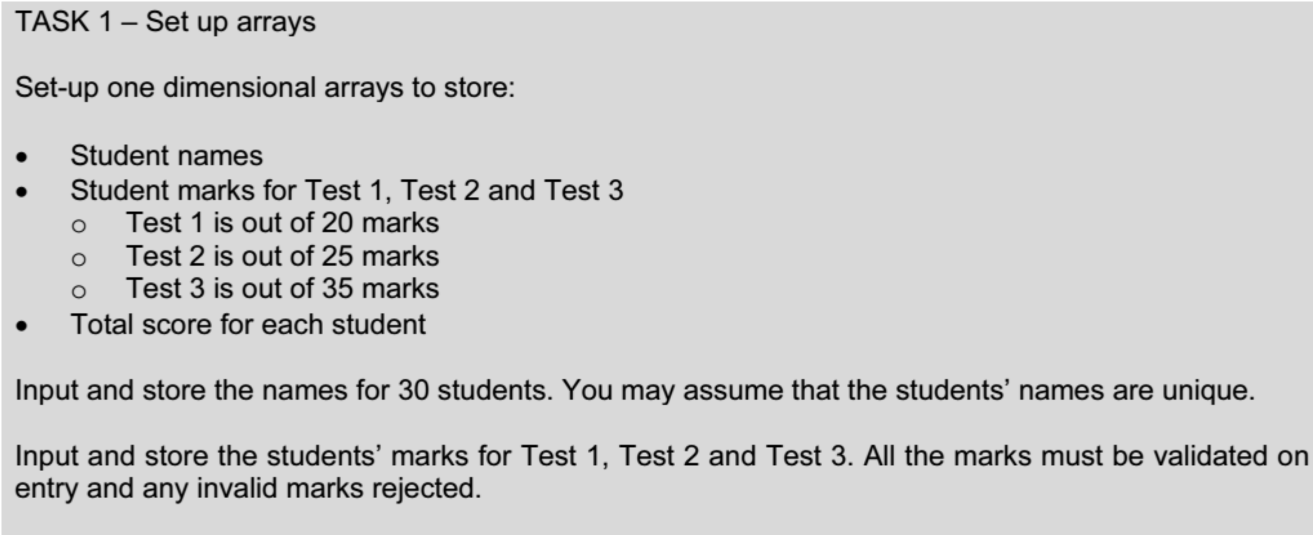
TASK 2 – Calculate

Calculate the total score for each student and store in the array.
Calculate the average total score for the whole class.

Output each student's name followed by their total score.
Output the average total score for the class.

TASK 3 – Select

Select the student with the highest total score and output their name and total score.



TASK 1 – Set up arrays

Set-up one dimensional arrays to store:

- Student names
- Student marks for Test 1, Test 2 and Test 3
  - Test 1 is out of 20 marks
  - Test 2 is out of 25 marks
  - Test 3 is out of 35 marks
- Total score for each student

Input and store the names for 30 students. You may assume that the students' names are unique.

Input and store the students' marks for Test 1, Test 2 and Test 3. All the marks must be validated on entry and any invalid marks rejected.

Data structure:

A**data structure** is a specialized format for organizing and storing **data**. General **data structure** types include the array, the file, the record, the table, and so on.

Data structure name	Data Type	Purpose
Name[30] or Name[1:30]	String	To input,store and display name of 30 students
Test1[30],[Test2[30], est3[30] Or Test1[1:30], Test2[1:30], Test3[1:30]	Real	To input and store marks in test1, test2 and test3
Student_Total[30] Or Student_Total[1:30]	Real	To calculate each students total marks

Variabels:

A variable is a memory location. It has a name (an identifier) that is associated with that location. The value associated with a variable name may change during program execution. For example, when the user is asked a question, for example, their age.

Variable Name	Data Type	Purpose
Count or Index	Integer	To count number of students from 1 to 30

Constatnts:

A constant is also a memory location. It has a name (an identifier) that is associated with that location. Data values that stay the same every time a program is executed are known as constants.

Constant Name	Data Type	Purpose
NumberOfStudents	Integer	To store the total number of students which remain same during execution of programs

Validation:

To reject if invalid marks
Test 1 out of 20
Test 2 out of 25
Test 3 out of 35

```
While Test1[Index] > 20 Do
        PRINT "Invalid marks, Enter marks out of 20 "
        READ Test1
Endwhile
```

Test Data:

To check corectness of pseudo code

Test Data Set	Purpose
18, 22, 32	To check input of Normal Data
25, 30, 40	To check rejection of Abnormal Data
20, 25, 35	To check input of Extreme Data

Pseudocode

```
//Declaration of variables
DECLAREName[30] : String
        DECLARETest1[30],  Test2[30],  Test3[30],  Student_Total[30]: Real
        DECLAREIndex : Integer
//Declaration of constant
        CONSTANT NumberOfStudents← 30
//Input and store name and marks of students
FORIndex ← 1 TO 30
        PRINT"Enter name of student "
        READ  Name[Index]
        PRINT"Enter marks in test 1 "
        READ Test1[Index]
                //Validation of test 1 marks out of 20
                WHILE Test1[Index] > 20 DO
                        PRINT "Invalid marks, Enter marks out of 20 "
                        READ Test1[Index]
                ENDWHILE
        PRINT"Enter marks in test 2 "
        READ Test2[Index]
                //Validation of test 2 marks out of 25
                WHILE Test1[Index] > 25DO
                        PRINT "Invalid marks, Enter marks out of 25 "
                        READ Test2[Index]
                ENDWHILE
        PRINT"Enter marks in test 3 "
        READ Test3[Index]
                //Validation of test 3 marks out of 35
                WHILE Test1[Index] >35DO
                        PRINT "Invalid marks, Enter marks out of 35 "
                        READ Test3[Index]
                ENDWHILE
        NEXT Index
```

Input and validation of marks in test 1,2 and 3Validation of test 1 marks out of 20, test 2 marks 25 and test 3 out of 35.

Visual Basic Code

```
Module Module1

Sub Main()
'Declaration of variables
Dim Name(30) AsString
Dim Test1(30), Test2(30), Test3(30), Student_Total(30) AsSingle
Dim Index AsInteger
'Declaration of constant
Const NumberOfStudents = 30
'Input and store name and marks of students

For Index = 1 To 30
            Console.Write("Enter name of student : ")
            Name(Index) = Console.ReadLine
            Console.Write("Enter marks in test 1 : ")

            Test1(Index) = Console.ReadLine
'Validation of test 1 marks out of 20
While Test1(Index) > 20
                Console.Write("Invalid marks, Enter marks out of 20 : ")
                Test1(Index) = Console.ReadLine
EndWhile

            Console.Write("Enter marks in test 2 : ")
            Test2(Index) = Console.ReadLine
'Validation of test 2 marks out of 25
While Test2(Index) > 25
                Console.Write("Invalid marks, Enter marks out of 25 : ")
                Test2(Index) = Console.ReadLine
EndWhile

            Console.Write("Enter marks in test 3 : ")
            Test3(Index) = Console.ReadLine
'Validation of test 3 marks out of 35
While Test3(Index) > 35
                Console.Write("Invalid marks, Enter marks out of 35 : ")
                Test3(Index) = Console.ReadLine
EndWhile

Next Index


EndSub

EndModule
```

TASK 2 – Calculate

Calculate the total score for each student and store in the array.
Calculate the average total score for the whole class.

Output each student's name followed by their total score.
Output the average total score for the class.

Variabels:

Variable Name	Data Type	Purpose
Class_Total	Real	To calculate class total
Class_Average	Real	To calculate class total average

Formulae:

To calulate each student total marks

```
Student_Total[Index]=Test1[Index]+Test2[Index] +Test3[Index]
```

To calulate class total

```
Class_Total= Class_Total+Student_Total[Index]
```

To calulate average total score of class

```
Class_Average=Class_Total/30
```

Initialization:

Class_Total is claculated using concept of TOTALLing (Total←Total + Number) so it is needed to be initialised with zero.

Class_Total←0

TraceTable

Index	Name	Test 1	Test 2	Test 3	Student_Total	Class_Total	Class_Average
					Student_Total[Index]=Test1[Index]+Test2[Index]+Test3[Index]	Class_Total= Class_Total+Student_Total[Index]	Class_Average=Class_Total/30
						0	
1	Ab	18	20	32	70	70	
2	CD	15	15	30	60	130	
30	EF	20	25	35	80	2100	70

Pseudocode of Task 2

```
//Declaration of variables
DECLARE Class_Total, Class_Average: Real

//Initialization of Class TotalThree Totals set to be zero
        Class_Total ← 0

//Calculation of student total, class total and class average

For Index = 1 To 30
     //To calulate each student total marks
          Student_Total[Index]=Test1[Index]+Test2[Index] +Test3[Index]
          //To calulate class  total
          Class_Total= Class_Total+Student_Total[Index]
     Next Index
     //To calulate average total score of class
     Class_Average=Class_Total/30

//Output each student’s name followed by their total
     For Index = 1 To 30
     PRINT "Student’s name  = " , Name[Index]
     PRINT "Student’s total = " , Student_Total[Index]
     Next Index

//Output average total score of class
     PRINT "Class Average  = " , Class_Average
```

Visual Basic Code of Task 2

```
'Declaration of variables
Dim Class_Total, Class_Average AsSingle

'Initialization of Class TotalThree Totals set to be zero
        Class_Total = 0

'Calculation of student total, class total and class average
For Index = 1 To 30
'To calulate each student total marks
            Student_Total(Index) = Test1(Index) + Test2(Index) + Test3(Index)
'To calulate class  total
            Class_Total = Class_Total + Student_Total(Index)
Next Index
'To calulate average total score of class
        Class_Average = Class_Total / 30

'Output each student’s name followed by their total
For Index = 1 To 30
            Console.WriteLine("Student’s name  = "& Name(Index))
            Console.WriteLine("Student’s total = "& Student_Total(Index))
Next Index

'Output average total score of class
        Console.WriteLine("Class Average  = "& Class_Average)
```

TASK 3 – Select

Select the student with the highest total score and output their name and total score.

Variabels:

Variable Name	Data Type	Purpose
Highest_Score	Real	To find out highest score in class
Best_Name	String	To store name of student with highest score

Initilisation:

```
        Highest Score is initialised with zero
        Highest_Score←0
```

Selection:

```
        To select highest score and best name
        If Student_Total[Index]> Then
                Highest_Score←Student_Total[Index]
                Best_Name←Name[Index]
        EndIf
```

Pseudocode

```
//Initialisation of highest score
        DECLAREHighest_Score: Real
        DECLARE Best_Name: String

//Initialization of Class TotalThree Totals set to be zero
        Highest_Score←0

//To find out highest score and best name

        For Index = 1 To 30
                IF Student_Total[Index]> Highest_Score THEN
                        Highest_Score←Student_Total[Index]
                        Best_Name←Name[Index]
                ENDIF
        Next Index
        //To calulate average total score of class
        Class_Average=Class_Total/30

//Output highest score and best name
        PRINT "Highest score of class  = " , Highest_Score
        PRINT "Best Student's name  = " , Best_Name
```

VB Code of Task 3

```
'Task 3
Dim Highest_Score AsSingle
Dim Best_Name AsString

'Initialization of Class TotalThree Totals set to be zero
        Highest_Score = 0

'To find out highest score and best name
For Index = 1 To 30
If Student_Total(Index) > Highest_Score Then
                Highest_Score = Student_Total(Index)
                Best_Name = Name(Index)
EndIf
Next Index
'To calulate average total score of class
        Class_Average = Class_Total / 30

'Output highest score and best name
        Console.WriteLine("Highest score of class  = "& Highest_Score)
        Console.WriteLine("Best Student’s name    = "& Best_Name)
```

Sample Questions

1 (a) All variables, constants and other identifiers should have meaningful names.

(i) Declare the array to store the students' names.

.... [1]

(ii) Declare the arrays to store each student's marks and total score.

....

.... [2]

(b) (i) Show the design of your algorithm to complete Task 1 and Task 2 using pseudo code,programming statements or a flowchart. Do not include any of the validation checks orinput prompts in your algorithm.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [8]

(ii) Comment on the efficiency of your design.

.... [1]

(c) Show two different sets of student data that you could use to check the validation used in Task 1. Explain why you chose each data set.

Set 1: ....

Reason for choice: ....

....

Set 2: ....

Reason for choice: ....

.... [2]

(d) (i) Explain how you select the student with the highest score (Task 3). You may include pseudo code or programming statements to help illustrate your explanation.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [5]

(ii) How does your program work when there is more than one student having the highest score? Explain using your method given in part (d)(i).

....

....

.... [1]

Marking Scheme

1 (a) (i) **Many correct answers, they must be meaningful. This is an example only.**

```
StudentNames[1:30]
```

[1]

(ii) **Many correct answers, they must be meaningful. This is an example only.**

```
StudentMarksTest1[1:30]
StudentMarksTest2[1:30]
StudentMarksTest3[1:30]  (1 mark)
StudentTotalScore[1:30]  (1 mark)
```

[2]

(b) (i)
- – outside loop zeroing total for loop (sum in example below)
- – loop for all students
- – input name and all test scores
- – in loop adding a student's total
- – storing the total
- – inside loop printing student's name and total
- – outside loop calculating class average
- – printing class average

sample algorithm:

```
Sum ← 0
FOR Count ← 1 TO 30
    INPUT Name
    StudentName[Count] ← Name
    INPUT Mark1, Mark2, Mark3
    StudentMarksTest1[Count] ← Mark1
    StudentMarksTest2[Count] ← Mark2
    StudentMarksTest3[Count] ← Mark3
    Total ← Mark1 + Mark2 + Mark3
    StudentTotalScore[Count] ← Total
    Sum ← Sum + Total
    PRINT StudentName[Count], StudentTotalScore[Count]
NEXT Count
ClassAverage = Sum/30
PRINT ClassAverage
```

[8]

(ii) any relevant comment with regards to efficient code (e.g. single loop) [1]

(c) **Many correct answers, these are examples only.**
1 mark per data set and reason

Set 1: 20, 25, 35
Reason: valid data to check that data on the upper bound of each range check is accepted

Set 2: 21, 26, 36
Reason: invalid data to check that data above the upper bound of each range check is rejected [2]

(d) **(i)** Maximum 5 marks **in total** for question part
Maximum 3 marks for algorithm

Description (max 3)

- set variable called HighestScore to zero and variable called BestName to dummy value
- loop 30 times to check each student's total score in turn
- check student's score against HighestScore
- if student's score > HighestScore then
- … replace value in HighestScore by student's score and store student's name in BestName
- output BestName and HighestScore outside the loop

Sample algorithm (max 3):

```
HighestScore ← 0
BestName ←  "xxxx"                                  (1 mark)
FOR Count ← 1 TO 30
   IF StudentTotalScore[Count] > HighestScore       (1 mark)
      THEN
         HighestScore ← StudentTotalScore[Count]
         BestName ← StudentName[Count]              (1 mark)
   ENDIF
NEXT Count                                          (1 mark)
PRINT BestName, HighestScore                        (1 mark)
```

If algorithm or program code only, then maximum 3 marks [5]

(ii) comment on which student(s)' name will be output
e.g. The first student with the highest score will be output [1]